

日 本 国 特 許 庁  
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日            2 0 0 4 年   2 月   2 日  
Date of Application:

出 願 番 号            特 願 2 0 0 4 - 0 2 5 6 7 7  
Application Number:  
[ST. 10/C]:            [ J P 2 0 0 4 - 0 2 5 6 7 7 ]

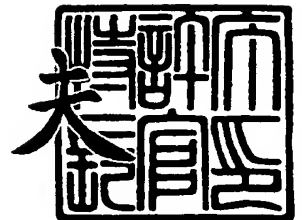
出   願   人            富 士 通 株 式 会 社  
Applicant(s):



2 0 0 4 年   2 月 1 6 日

特許庁長官  
Commissioner,  
Japan Patent Office

今 井 康 夫



【書類名】 特許願  
【整理番号】 0352336  
【提出日】 平成16年 2月 2日  
【あて先】 特許庁長官殿  
【国際特許分類】 G06F 19/00  
【発明者】  
    【住所又は居所】 神奈川県川崎市中原区上小田中 4 丁目 1 番 1 号 富士通株式会社  
                                内  
    【氏名】 土河原 信雄  
【特許出願人】  
    【識別番号】 000005223  
    【氏名又は名称】 富士通株式会社  
【代理人】  
    【識別番号】 100103528  
    【弁理士】  
    【氏名又は名称】 原田 一男  
【先の出願に基づく優先権主張】  
    【出願番号】 特願2003-283314  
    【出願日】 平成15年 7月31日  
【手数料の表示】  
    【予納台帳番号】 076762  
    【納付金額】 21,000円  
【提出物件の目録】  
    【物件名】 特許請求の範囲 1  
    【物件名】 明細書 1  
    【物件名】 図面 1  
    【物件名】 要約書 1  
    【包括委任状番号】 9909129

**【書類名】 特許請求の範囲****【請求項 1】**

XML データ格納部に格納され且つ帳票画面に対応する XML データを解析し、前記 XML データに含まれるタグに対応する業務クラスを特定する業務クラス特定ステップと、  
生成すべき業務クラスの項目が登録された業務クラス管理部を参照して、特定された前記業務クラスが未登録であるか判断するステップと、

特定された前記業務クラスの項目が未登録である場合には、前記業務クラス管理部に当該特定された前記業務クラスの項目を登録するステップと、  
を含み、コンピュータにより実行される情報処理方法。

**【請求項 2】**

特定された前記業務クラスの項目が未登録である場合には、当該特定された前記業務クラスののための雛型ソースプログラム・データを生成し、ソースプログラム格納部に格納するステップ

をさらに含む請求項 1 記載の情報処理方法。

**【請求項 3】**

帳票画面に対する入力又は選択データに対応するタグを含む XML データをメモリに格納し、当該 XML データに含まれるタグに対応し且つ当該タグに関連する処理を実施するためのプログラムである業務クラスを特定する業務クラス特定ステップと、

予め定義され且つメモリにロードされた業務クラスの中で、特定された前記業務クラスを呼び出す呼出ステップと、

を含み、コンピュータにより実行される情報処理方法。

**【請求項 4】**

前記帳票画面に係る帳票に対応して当該帳票全体の処理シーケンスが規定されておらず、呼び出された前記業務クラスの処理にて当該帳票に対する処理が完了するように、前記業務クラスの各々が構成されていることを特徴とする請求項 3 記載の情報処理方法。

**【請求項 5】**

帳票画面に対する入力又は選択データに対応するタグを含む XML データをメモリに格納し、当該 XML データに含まれるタグに対応し且つ当該タグに関連する処理を実施するためのプログラムである業務クラスを特定する業務クラス特定ステップと、

予め定義され且つメモリにロードされた業務クラスの中で、特定された前記業務クラスを呼び出す呼出ステップと、

をコンピュータに実行させるプログラム。

## 【書類名】明細書

【発明の名称】XMLドリブンアーキテクチャにおける情報処理方法及びプログラム

## 【技術分野】

## 【0001】

本発明は、XML (eXtensible Markup Language) をベースとしたプログラム開発技術及びXMLをベースとした情報処理技術に関する。

## 【背景技術】

## 【0002】

従来では、画面及び入出力データ、プログラムの機能分担、プログラムの構造、データベースのデータ項目などを一元的な観点で考慮した上で、業務プログラムの開発がなされていた。このため、1つの業務プログラムの同時並列的な設計は不可能であり、業務プログラムの規模が大きくなると開発期間が長期化することも多かった。また、仕様の変更・追加を行う際には、全面的な見直しが必要となり、検討時間の増加、及び修正時間の増加を生じていた。

## 【0003】

このような問題を解決すべく特開平9-198240号公報は、画面の設計時に画面上の入力フィールドに入力される文字数や文字の入出力定義体を生成し、さらにモックアップに用いられていたプログラムエントリの工程をそのままプログラムの製造工程に自動継承させて、各業務ルーチンの設計を容易に行えるようにする技術を開示している。

【特許文献1】特開平9-198240号公報

## 【発明の開示】

## 【発明が解決しようとする課題】

## 【0004】

しかしながら、特許文献1に開示された技術では、入出力定義体の自動生成やプログラムエントリの工程の自動継承は容易になったが、自動継承された設計情報を基に業務個別ルーチンを作成しなければならず、プログラムを同時並行して作成する場合、プログラムの作成漏れなどが生じてしまうことがある。また、プログラム実行の際に行われる入力データの受け渡しは、設計時に決定したIDにより行われているので、後に修正などする場合には、その設計者以外には分かりにくい構成となっている。

## 【0005】

従って、本発明の目的は、より開発期間を短縮でき且つより品質の高い業務プログラムの開発を可能にするための技術を提供することである。

## 【0006】

また本発明の他の目的は、本発明に係るプログラム開発技術により作成された業務プログラムを実行するための基盤技術を提供することである。

## 【課題を解決するための手段】

## 【0007】

本発明の第1の態様に係る情報処理方法は、XMLデータ格納部に格納され且つ帳票画面に対応するXMLデータを解析し、XMLデータに含まれるタグに対応する業務クラスを特定する業務クラス特定ステップと、生成すべき業務クラスの項目が登録された業務クラス管理部を参照して、特定された業務クラスの項目が未登録であるか判断するステップと、特定された業務クラスの項目が未登録である場合には、業務クラス管理部に当該特定された業務クラスの項目を登録するステップとを含む。

## 【0008】

帳票画面に対応するXMLデータのタグをベースに業務クラスを作成する際に、業務クラス管理部において、生成すべき業務クラスの項目を管理するため、重複する業務クラスを作成したり、不足が生じたりすることなくプログラム開発を行うことができる。なお、XMLのタグに対応する業務クラスを作成すればよいので、全体を把握せずとも業務クラスを作成でき、業務プログラムの品質向上が容易で、開発期間の短縮も図ることができる。

**【0009】**

また、特定された業務クラスの項目が未登録である場合には、当該特定された業務クラスのための雛型ソースプログラム・データ（例えばファイル）を生成し、ソースプログラム格納部に格納するステップをさらに含むようにしてもよい。プログラム作成者は、生成された雛型ソースプログラム・データを用いて所定の機能を実現するためのプログラムを作成すればよく、過不足無く業務クラスを用意できる。

**【0010】**

さらに、帳票項目についてのデータを格納するためのオブジェクトの雛型ソースファイルについても、業務クラスと同様な処理にて生成するようにしても良い。

**【0011】**

本発明の第2の態様に係る情報処理方法は、例えば上記のような方法を用いて作成された業務クラスの実行方式に関する情報処理方法であって、帳票画面に対する入力又は選択データに対応するタグを含むXMLデータをメモリに格納し、当該XMLデータに含まれるタグに対応し且つ当該タグに関連する処理を実施するためのプログラムである業務クラスを特定する業務クラス特定ステップと、予め定義され且つメモリにロードされた業務クラスの中で、特定された業務クラスを呼び出す呼出ステップとを含む。

**【0012】**

業務クラスは元々XMLデータに含まれるタグに対応して作成されているため、特定の選択又は入力データを含むXMLデータが入力された場合には、当該XMLデータに含まれるタグに対応する業務クラスを呼び出すことにより、特定の選択又は入力データを処理するものである。

**【0013】**

また、帳票画面に係る帳票に対応して当該帳票全体の処理シーケンスが規定されておらず、呼び出された前記業務クラスの処理にて当該帳票に対する処理が完了するように、業務クラスの各々が構成されるようにしてもよい。このように、XMLデータのタグに対応するクラスにより必要な業務ロジックが実装されれば、システム開発の効率が向上する。

なお、タグに対応する業務クラスが存在しない場合には、不適切なXMLデータであるから、エラーを通知するような構成であってもよい。

**【0014】**

また、帳票画面への入力又は選択データを帳票画面を表示した装置から受信し、当該帳票画面への入力又は選択データに対応するタグを含むXMLデータを生成するステップをさらに実行するようにしても良い。

**【0015】**

さらに、帳票画面に対する入力又は選択データを、予め定義され且つメモリにロードされた帳票項目格納オブジェクトに出力し、当該帳票項目格納オブジェクトにより帳票画面に対する入力又は選択データをメモリに格納するステップと、呼び出された業務クラスと帳票項目格納オブジェクトとがデータをやり取りすることにより処理を実施する処理実施ステップとをさらに含むようにしても良い。

**【0016】**

さらに、呼び出された業務クラスにより出力要求を受けた場合、帳票項目格納オブジェクトに保持されたデータを用いて出力用XMLデータを生成し、メモリに格納するステップと、メモリに格納された出力用XMLデータを帳票画面を表示した装置に出力するステップとをさらに実行するようにしても良い。

**【0017】**

また、本発明に係る情報処理方法をコンピュータに実行させるためのプログラムを作成することも可能であって、当該プログラムは、例えばフレキシブル・ディスク、CD-ROM、光磁気ディスク、半導体メモリ、ハードディスク等の記憶媒体又は記憶装置に格納される。プログラム等は、ネットワークを介してデジタル信号として配信される場合もある。また、処理途中のデータについては、コンピュータのメインメモリ等の記憶装置に一時的に保管される。

**【発明の効果】****【0018】**

本発明によれば、より開発期間を短縮でき且つより品質の高い業務プログラムの開発を行うことができる。

**【0019】**

また別の側面として、本発明に係るプログラム開発技術により作成された業務プログラムを実行するための基盤技術を提供できる。

**【発明を実施するための最良の形態】****【0020】****1. 業務クラス（プログラム）の作成**

図1に、業務クラスの作成時における、本発明の一実施の形態に係るコンピュータ・システムの機能ブロック図を示す。コンピュータ・システム100は、開発中のシステムにおいて表示される画面の構成を行い、当該画面に対応するHTML (Hyper Text Markup Language) ファイルなどを生成する画面作成部1と、画面作成部1で生成されたHTMLファイルを格納するHTMLファイル格納部3と、HTMLファイル格納部3に格納されたHTMLファイルを解析し、所定のルールに従ってXMLデータに変換するXML変換部5と、XML変換部5により生成されたXMLデータ等を格納するXMLデータ格納部7と、XMLデータ格納部7に格納されたXMLデータを解析して、共通インターフェースファイル群11の少なくとも一部及び雛型業務ソースファイル群13を生成するインターフェースマネージャ9と、インターフェースマネージャ9により参照及び更新される業務クラス管理ファイル15とを含む。

**【0021】**

画面作成部1は、一般に販売されているホームページエディタなどであり、ユーザがHTMLのタグを記述せずに画面構成を行うとHTMLファイルなどを生成するものである。例えば図1に示すように、銀行名、支店名、口座種別、口座番号及び取引金額の入力欄を含む入金取引の画面1aを構成し、対応するHTMLファイル等が生成されるものとする。生成されたHTMLファイルはHTMLファイル格納部3に格納される。

**【0022】**

XML変換部5は、一般に販売されているXML SPY (ICON社の商標) 等のXML文書作成支援ツールであり、HTMLファイル格納部3に格納されたHTMLファイルを読み込み、XMLデータ、スタイルシート(XSL (eXtensible Style Language) データ)、スキーマを生成し、XMLデータ格納部7に格納するものである。なお、以下では、本実施の形態のポイントの把握を容易にするため、XMLデータについての説明を中心に行っていくが、XSLデータには、XMLデータを画面表示するためのデータ(書式等)が含まれる。またスキーマには、XMLデータの妥当性検査を行うためのデータ(タグ構造等)が含まれる。図1に示すように、画面1aのためのHTMLファイルから、入金取引入力というタグ(開始及び終了タグ。以下同じ。)と、銀行名というタグと、支店名というタグと、口座種別というタグと、口座番号というタグと、取引金額というタグとを含むXMLデータ5aが生成される。また併せて、XMLデータ5aに対応するXSLデータ及びスキーマが生成される。

**【0023】**

雛型業務ソースファイル群13には、インターフェースマネージャ9により生成され且つXMLデータに含まれるタグに対応する雛型ソースプログラムのファイルが含まれる。図1に示すように、インターフェースマネージャ9がXMLデータ5aを処理すると、入金取引入力という伝票の前処理を行うための「入金取引入力前処理」という業務クラス131と、銀行名タグに対応する「銀行名」という業務クラス132と、支店名タグに対応する「支店名」という業務クラス133と、口座種別タグに対応する「口座種別」という業務クラス134と、口座番号タグに対応する「口座番号」という業務クラス135と、取引金額タグに対応する「取引金額」という業務クラス136と、入金取引入力という伝票処理に対応する「入金取引入力」という業務クラス137と、入金取引入力という伝票

の後処理を行うための「入金取引入力後処理」という業務クラス 1 3 8 とが、雛型ソースファイルとして生成される。雛型ソースファイルは、業務クラスとしての基本的なソースコードを含むだけであって、後で業務ロジックを例えばメソッドとしてコーディングする必要がある。なお、雛型ソースファイルは、空であってもよい。

#### 【 0 0 2 4 】

以下に口座番号という業務クラスの雛型ソースファイルの一例を示す。なお、この具体的内容については J a v a (Sun Microsystems社の登録商標) 言語によって記述された一般的なものであるから、詳細な説明については省略する。

#### 【 0 0 2 5 】

[表 1]

```
package com.fff.xxxx.usr;

import com.fff.xxxx.common.*;
import com.fff.xxxx.util.*;
import com.fff.xxxx.processor.*;
//クラス：USR_口座番号
public class USR_口座番号 implements XxxxUserProcessLibrary
{
    //コンストラクタ
    public USR_口座番号()
    {
        //ここにコンストラクタでの処理内容を記述してください。
    }

    //プロセス実行関数
    public void process(XxxxInterface rinterface) throws Exception
    {
        //オブジェクトプールから自領域を取得する。
        String 口座番号 = (String)rinterface.getValue_XML("口座番号");

        //ここに処理単位毎のプログラムを記述してください。
    }
}
```

#### 【 0 0 2 6 】

また、共通インターフェースファイル群 1 1 には、伝票項目域 1 1 1 と、プロセッサ予約域 1 1 2 と、クラス間インターフェース領域 1 1 3 と、アプリケーション定数域 1 1 5 と、データベース (DB) 情報域 1 1 7 と、ユーザがエディタ等を用いて作成する初期設定ファイル 1 1 9 とが含まれる。初期設定ファイル 1 1 9 を除くこれらの領域は、例えば複数の業務クラスにより共通に用いられるオブジェクトの雛型ソースプログラム (以下、共通インターフェース雛型ソースファイルと呼ぶ。) が格納される領域である。共通インターフェース雛型ソースファイルは、上で述べた業務クラスの雛型ソースファイル同様、基本的なソースコードを含むだけであって、後でロジックを例えばメソッドとしてコーディングする必要がある。但し、共通インターフェース雛型ソースファイルは、空であってもよい。

#### 【 0 0 2 7 】

伝票項目域 1 1 1 には、伝票項目データを格納するためのオブジェクトの共通インターフェース雛型ソースファイルが格納される。伝票項目域 1 1 1 に格納される共通インターフェース雛型ソースファイルは、インターフェースマネージャ 9 によって XML データに基づき生成される。例えば、XML データに含まれる伝票項目の各々に対応して共通インターフェース雛型ソースファイルが生成される。

## 【 0 0 2 8 】

以下に、口座番号という伝票項目に係る共通インターフェース雛型ソースファイルの一例を示す。但し、この具体的内容は、J a v a 言語によって記述された一般的なものであるから、詳細な説明については省略する。

## 【 0 0 2 9 】

[表 2]

```
package com.fff.xxxx.pool;

import com.fff.xxxx.*;

import com.fff.xxxx.common.*;

import java.util.*;
////LOG出力用
/*LOG*/import com.fff.OPERAPARTS.com.SSCLG00C;

/** クラス：com_口座番号 */
public class com_口座番号 implements XxxxCommonInLibrary {

    /** ハッシュの何番目に格納するかのフラグ */
    public int flagGet = 0;
    /** ハッシュの何番目から取り出したかのフラグ */
    public int flagSet = 0;
    /** エラーハッシュの何番目に格納するかのフラグ */
    public int errflagGet = 0;
    /** エラーハッシュの何番目から取り出したかのフラグ */
    public int errflagSet = 0;
    /** 子タグクラスを格納するハッシュテーブル*/
    private Hashtable childtags = new Hashtable();

    /** 値を格納するハッシュテーブル*/
    private Hashtable values = new Hashtable();

    /** エラーコードを格納するハッシュテーブル*/
    private Hashtable errors = new Hashtable();

    /** 属性ハッシュを管理するハッシュテーブル*/
    private Hashtable attributes = new Hashtable();

    /** 属性値を格納するハッシュテーブル*/
    ////LOG出力用
    /*LOG*/ private SSCLG00C ewkoSsclg00c;
    /** コンストラクタ */
    public com_口座番号() {

        //ここにコンストラクタでの処理内容を記述してください。

        ////LOG出力用
        /*LOG*/ ewkoSsclg00c = new SSCLG00C(this.getClass().getName());
```



```
{

/** プロセス実行関数 */
public void process(XxxxInterface rinterface) {

    ewkoSscIlg00c.mvLogWrite(3,"プロセス呼び出しクラス名      : com_口座番号.pr
rocess()");
}

/** setメソッド */
public void set(Object val) {
    String numstr = String.valueOf(flagGet);
    values.put(numstr, val);
    flagGet += 1;
}
/** getメソッド */
public Object get(int key) {
    String numstr = String.valueOf(key);
    return values.get(numstr);
}
/** getメソッド最初の項目を取得 */
public Object get() {
    String numstr = String.valueOf(flagGet - 1);
    return values.get(numstr);
}
/** valueテーブルの中身をクリアする*/
public void clearValue() {
    attributes.clear();
    values.clear();
    clearStatus();
    flagGet = 0;
}
//エラーコードを格納する
public void setStatus(int value) {
    String numstr = String.valueOf(errflagGet);
    errors.put(numstr, new Integer(value));
    errflagGet += 1;
}
//エラーコードを取得する
public int getStatus( String key) {
    return ((Integer)errors.get(key)).intValue();
}
//エラーコードを取得する
public int getStatus() {
    String numstr = String.valueOf(errflagGet - 1);
    return ((Integer)errors.get(numstr)).intValue();
}
/** エラーコード格納テーブルの中身をクリアする*/
```

```

public void clearStatus() {
    errors.clear();
    errflagGet = 0;
    setStatus(0);
}
//繰り返し数を取得する
public int getNumber() {
    return values.size();
}
//属性の値を取得
public Object getAttr(int num, String attsname) {
    String numstr = String.valueOf(num);
    Hashtable attshash = (Hashtable)attributes.get(attsname);
    return attshash.get(numstr);
}
//属性に値をセット
public void setAttr(int num, String attsname, Object attsVal) {
    String numstr = String.valueOf(num);
    Hashtable attshash = (Hashtable)attributes.get(attsname);
    attshash.put(numstr, attsVal);
}
//属性名を戻す
public Enumeration getAttrNames() {
    return attributes.keys();
}
//属性名から、属性値の入ったHashを返す
public Hashtable getAttrs(String attsname) {
    return (Hashtable)attributes.get(attsname);
}
}

```

#### 【0030】

プロセッサ予約域 1 1 2 には、業務クラスの実行処理に関わるプロセッサ（図 6 のプロセッサ 2 0 7 1）の制御に用いるデータを格納するためのオブジェクトの共通インターフェース雛型ソースファイルが格納される。プロセッサ予約域 1 1 2 に格納される共通インターフェース雛型ソースファイルは、インターフェースマネージャ 9 が有する画面入力機能（ユーザインターフェース）を介して設定されたデータ項目に応じて、インターフェースマネージャ 9 により生成され、プロセッサ予約域 1 1 2 に格納される。画面入力機能とは、ユーザからのデータ入力を受け付ける画面を表示させ、入力されたデータに基づき項目を設定する機能である。

#### 【0031】

クラス間インターフェース領域 1 1 3 には、業務クラス間のインターフェース項目に関するデータを格納するためのオブジェクトの共通インターフェース雛型ソースファイルが格納される。なお、業務クラス間のインターフェース項目とは、例えば複数の業務クラス間におけるデータの受け渡しを行うために用いる項目である。クラス間インターフェース領域 1 1 3 に格納される共通インターフェース雛型ソースファイルは、プロセッサ予約域 1 1 2 に格納される共通インターフェース雛型ソースファイル同様、インターフェースマネージャ 9 が有する画面入力機能を介して設定されたデータ項目に応じて、インターフェースマネージャ 9 により生成され、プロセッサ予約域 1 1 2 に格納される。

#### 【0032】

アプリケーション定数域 1 1 5 には、業務クラスが参照する定数項目データを格納する

ためのオブジェクトの共通インターフェース雛型ソースファイルが格納される。なお、業務クラスが参照する定数項目とは、初期設定ファイル 119 において規定されているデータに基づき設定される項目であり、例えばプログラム実行可能時間帯等の業務ルールに関する項目である。アプリケーション定数域 115 に格納される共通インターフェース雛型ソースファイルは、プロセッサ予約域 112 やクラス間インターフェース領域 113 に格納される共通インターフェース雛型ソースファイル同様、インターフェースマネージャ 9 が有する画面入力機能を介して設定されたデータ項目に応じて、インターフェースマネージャ 9 により生成され、アプリケーション定数域 115 に格納される。

#### 【0033】

DB情報域 117 には、DBのテーブル定義項目データを格納するためのオブジェクトの共通インターフェース雛型ソースファイルが格納される。DB情報域 117 に格納される共通インターフェース雛型ソースファイルは、伝票項目域 111 に格納される共通インターフェース雛型ソースファイル同様、インターフェースマネージャ 9 によってXMLデータに基づき生成される。例えば、XMLデータに含まれる伝票項目の各々に対応して共通インターフェース雛型ソースファイルが生成される。

#### 【0034】

初期設定ファイル 119 は、上で述べた定数項目データを含むファイルであり、例えばXML形式のデータである。業務クラスの実行処理において、このファイルは、業務クラスによる処理を実行する際に読み込まれ、対応するオブジェクトに定数項目データが保持される。

#### 【0035】

業務クラス管理ファイル 15 は、例えば図 2 に示すようなデータを含む。図 2 には、XML ファイルを指定していないXMLデータ（業務クラス管理ファイル 15）をXML対応のブラウザで表示した場合の画面例が示されている。図 2 の例では、資産管理タグにより、業務クラス管理ファイル 15 により開発するプログラムにおいて必要とされる資産の管理を行っていることを表している。また、本実施の形態では業務クラスだけではなく、共通インターフェースファイル群 11 についても管理するようになっており、オブジェクトプール・タグが設けられている。ここでオブジェクトプールとは、共通インターフェース雛型ソースファイルに基づき実行可能なクラスが生成されている場合に、処理実行時に当該クラスがロードされるメモリ領域を指す。オブジェクトプール・タグ配下には、プロセッサ予約域 112 に対応するプロセッサ予約域タグと、伝票項目域 111 に対応する伝票項目域タグと、DB情報域 117 に対応するDB情報域タグと、アプリケーション定数域 115 に対応するアプリ定数項目域タグと、クラス間インターフェース領域 113 に対応するクラス間インターフェース項目域タグとが設けられている。図 4 の例では、「+」印でプロセッサ予約域タグ及びクラス間インターフェース項目域タグの配下には既に定義された項目があることを示している。一方、DB項目域タグ及びアプリ定数項目域タグの配下にはまだ定義されている項目はない。ここで「項目が定義されている」とは、項目に対応する共通インターフェース雛型ソースファイルが生成され、共通インターフェースファイル群 11 のいずれかの領域に格納されているということを指す。

#### 【0036】

いずれの領域に対応する共通インターフェース雛型ソースファイルについてもインターフェースマネージャ 9 により各領域に格納するようになっており、そして、インターフェースマネージャ 9 は共通インターフェース雛型ソースファイルをいずれかの領域に格納すると、当該領域に対応するタグの配下に、共通インターフェース雛型ソースファイルに対応する項目を登録する。

#### 【0037】

図 2 では、伝票項目域タグ配下には既に定義された項目が存在しており、それが具体的に示されている。図 1 の画面 1a に対応して、「入金取引入力」タグ、「取引金額」タグ、「口座番号」タグ、「口座種別」タグ、「支店名」タグ及び「銀行名」タグが設けられている。「入金取引入力」は伝票に対応するため「ルート判定」という属性が規定されて

いる。また各タグでは、登録日時という属性が規定されている。

#### 【0038】

また、図2の例では、生成すべき業務クラス（すなわち生成した雛型業務ソースファイル）が、雛型業務クラス・タグの配下に規定されている。ここでは、取引金額タグ、口座番号タグ、口座種別タグ、支店名タグ、銀行名タグ、入金取引入力前処理タグ、入金取引入力後処理タグ及び入金取引入力タグとが含まれる。なお各タグには、登録日時という属性が規定されている。またタグ名の前にUSRとあるのは、ユーザにより用意されるべきクラスであることを示すものである。

#### 【0039】

次に図3を用いて図1に示したコンピュータ・システム100の処理フローを説明する。まず本コンピュータ・システム100のユーザは、現行の伝票、帳票、画面構成を検討すると共に、これから開発する新たなコンピュータ・システム用に追加又は変更すべき機能を把握し、画面作成部1により新たなコンピュータ・システム用の画面構成を行う。これにより、画面作成部1は、HTMLファイルなどを生成し、HTMLファイル格納部3に格納する（ステップS1）。なお、ツールなどを用いたりせずともHTMLファイルの作成は可能であるため、本ステップについては点線ブロックで示している。また、本ステップの処理を行うためのツールについては、インターフェースマネージャ9とは別にユーザが用意することも可能である。

#### 【0040】

次に、XML変換部5は、HTMLファイル格納部3に格納されたHTMLファイルを読み出して、XMLデータを生成し、XMLデータ格納部7に格納する（ステップS3）。この際、XMLデータに対応するXSLデータ及びスキーマを併せて生成し、XMLデータ格納部7に格納しておく。なお、本ステップの処理を行うためのツールについても、インターフェースマネージャ9とは別にユーザが用意することも可能であり、点線ブロックで表されている。次に、インターフェースマネージャ9は、プロセス初期処理及び終了処理クラスのための雛型ソースプログラムを生成し、例えば雛型ソースファイル群13を格納する記憶装置に格納する（ステップS5）。プロセス初期処理及び終了処理クラスのためのソースプログラムについても雛型であり、プロセス初期処理及び終了処理のための具体的な機能を実現するためのプログラムは、後にコーディングされる。

#### 【0041】

そして、インターフェースマネージャ9は、XMLデータ格納部7を参照し、XMLデータの未処理タグを1つ選択する（ステップS7）。そして、業務クラス管理ファイル15の雛型業務クラス・タグの配下を参照して、ステップS7で選択されたタグが登録されていないか確認する（ステップS9及びS11）。業務クラス管理ファイル15の雛型業務クラス・タグの配下に既にタグが登録されているということは、既に雛型業務ソースファイルも生成されているということであり、再度生成すると重複を生じてしまうために確認する。もし、既に登録されていると判断された場合には、ステップS23に移行する。一方、未登録であることが確認できれば、当該タグが先頭タグであるか判断する（ステップS13）。すなわち、XMLデータにおいて伝票を特定するタグであるか判断する。先頭タグであると判断された場合には、当該伝票の前処理、後処理及び伝票クラスの雛型ソースプログラムを生成し、例えば雛型ソースファイル群13を格納する記憶装置に格納する（ステップS15）。そしてステップS21に移行する。一方、先頭タグではないと判断された場合には、当該タグが終了タグであるか判断する（ステップS17）。例えば伝票を特定するタグの終了タグであるかを判断する。もし、終了タグであると判断された場合にはステップS23に遷移する。一方、終了タグではないと判断された場合には、通常のタグであるので、当該タグに対応する業務クラスの雛型ソースプログラムを生成し、例えば雛型ソースファイル群13を格納する記憶装置に格納する（ステップS19）。上でも述べたが、雛型ソースプログラムは、基本的なソースコードのみが含まれるだけのものであり、そのみで業務クラスが完成することは無い。

#### 【0042】

ステップS15又はステップS19の後に、タグ名を業務クラス管理ファイル15の雛型業務クラス・タグの配下に登録する(ステップS21)。これにより重複する業務クラスを生成することが無くなり、プログラム開発における業務効率化及びプログラムの品質向上に繋がる。そして、全てのタグを処理したか判断する(ステップS23)。もし未処理タグが存在する場合にはステップS7に戻る。未処理タグが存在していない場合には処理を終了する。

#### 【0043】

なお、上の処理では先頭タグに対応して、伝票の前処理、後処理及び伝票処理の雛型ソースプログラムを生成するようにしているが、終了タグに対応して上記雛型ソースプログラムを生成するようにしても良い。

#### 【0044】

そして、インターフェースマネージャ9は、共通インターフェースファイル群11の各領域に対応する共通インターフェース雛型ソースファイルを生成し、共通インターフェースファイル群11の各領域に格納する(ステップS25)。伝票項目域111及びDB情報域117に格納される共通インターフェース雛型ソースファイルは、伝票項目に対応するため、上で述べた業務クラスの雛型ソースプログラムの生成処理と同じような処理フローにて生成される。すなわち、XMLデータに含まれるタグのうち伝票項目に対応する未処理タグを1つ選択し、業務クラス管理ファイル15においてオブジェクトプール・タグの配下に設定されている伝票項目域タグ及びDB情報域タグの配下を参照して、当該タグが未登録であるか確認する。未登録であれば、当該タグに対応する共通インターフェースファイル雛型ソースファイルを伝票項目域111及びDB情報域117の各々について生成し、さらに業務クラス管理ファイル15のオブジェクトプール・タグの下にタグを登録する。但し、業務クラス管理ファイル15に未登録であることを確認の後に伝票項目に対応するタグであるか確認するような処理であってもよい。

#### 【0045】

また、プロセッサ予約域112、クラス間インターフェース領域113及びアプリケーション定数域115に格納される共通インターフェース雛型ソースファイルは、インターフェースマネージャ9が有する画面入力機能により受け付けた、ユーザからの入力データに基づき生成される。インターフェースマネージャ9は、共通インターフェース雛型ソースファイルを生成すると、対応するタグを、業務クラス管理ファイル15の各領域に対応するタグの配下に登録する。

#### 【0046】

なお、上では雛型ソースプログラム(基本的にはそのためのファイル)を生成するような処理フローを示したが、必ずしも雛型ソースプログラムや共通インターフェース雛型ソースファイルを生成する必要は無い。生成すべき業務クラスや共通インターフェースファイルが業務クラス管理ファイル15に登録されていれば十分な場合もある。このような場合には、業務クラス管理ファイル15の項目に対応してソースプログラムをコーディングすればよい。そしてコーディングが終了した時点で、再度業務クラス管理ファイル15の各項目とマッチングをとればよい。

#### 【0047】

このようにして生成された雛型ソースプログラム及び共通インターフェース雛型ソースファイルについては、業務ロジックに適合するようなコーディングが実施される。すなわち、雛型ソースプログラムは、当該雛型ソースプログラムに共通の事項のみが入力済みとなっており、それぞれにおいて実現すべき機能については、プログラマが別途コーディングする。また、共通インターフェース雛型ソースファイルについても同様に、当該共通インターフェース雛型ソースファイルに共通の事項のみが入力済みとなっており、それぞれにおいて実現すべき機能については、プログラマが別途コーディングする。コーディングされたソース・プログラムはコンパイルされ、実行可能なプログラム・モジュールが生成される。

#### 【0048】

この際、特定の伝票に対応するXMLデータをステップS15及びステップS19で生成することが決定された業務クラスと当該業務クラスが必要とするオブジェクトプール内のオブジェクトのみで処理できるように、コーディングする。これは以下でも述べるが、特定の伝票に対応するXMLデータを処理する際には、XMLデータに含まれるタグをベースに、ステップS15及びステップS19で生成することが決定された業務クラスのみがコールされるためである。なお、オブジェクトプールにおけるオブジェクトは、業務クラスにより呼び出されるが、業務クラスによる処理をサポートするものであって、処理の主体は業務クラスにある。従来のようにXMLデータを予め規定された順番でシーケンシャルに処理するためのメイン制御ロジックを用意する必要は無く、業務クラスによりXMLデータのタグで規定される各オブジェクトを実行する。コーディングされたソース・プログラムはコンパイルされ、実行可能なプログラム・モジュールが生成される。

#### 【0049】

なお、共通インターフェースファイル群11の各領域に格納された共通インターフェース雛型ソースファイルについては、図4に示すようなオブジェクトを実現すべくコーディングがなされる。なお、図4に示すオブジェクトの構造はオブジェクトプール内では全て同じであり、例えば特定の雛型オブジェクト500には、データ部510とメソッド部520とが含まれている。データ部510には、データ格納数領域511と、1又は複数のデータ領域512と、1又は複数のデータ状態値領域513と、1又は複数の第1属性データ領域514と、・・・、1又は複数の第n（nは例えば自然数）属性データ領域515と、1又は複数の第1属性状態値領域516と、・・・、1又は複数の第n属性状態値領域517とが含まれている。また、メソッド部520には、項目参照部521と項目更新部522とデータ状態値参照部523とデータ状態値更新部524と項目初期化部525と属性参照部526と属性更新部527とデータ格納数取得部528とが含まれている。

#### 【0050】

このようなオブジェクトプールにおける各オブジェクトを使用するために、例えば図5（a）乃至図5（h）に示すようなコードを、業務クラスの雛型ソースプログラムに追加する。図5（a）には、共通インターフェースファイル群11の領域別に、図4に示した項目参照部521を呼び出すためのコードの例が示されている。まず1行目には、伝票項目域111に対応するオブジェクトの項目参照部521を呼び出すためのコードの例として、「オブジェクトプールインスタンス変数. getValue（伝票項目名, 配列番号）」というコードが示されている。例えばプログラマは「オブジェクトプールインスタンス変数」部分、「伝票項目名」部分及び「配列番号」部分に項目データを特定するための値を設定することにより、適切なオブジェクトによる処理を規定することができる。なお、本実施の形態においては、配列番号が0の場合、先頭レコードを指すようになっており、以下の説明でも同様である。伝票項目域111に対応するオブジェクトの項目参照部521は、このようなコードに従い、項目データを参照して値を返す。

#### 【0051】

2行目以降には、順に、DB情報域117、クラス間インターフェース領域113、アプリケーション定数域115及びプロセッサ予約域112の各々について、その領域に対応するオブジェクトの項目参照部521を呼び出すためのコードの例が示されている。例えばプログラマは、このようなコードを業務クラスの雛型ソースプログラムに追加することにより、オブジェクトプールに格納されている項目データを参照する処理を規定することができる。

#### 【0052】

図5（b）には、共通インターフェースファイル群11の領域別に、図4に示した項目更新部522を呼び出すためのコードの例が示されている。まず1行目には、伝票項目域111に対応するオブジェクトの項目更新部522を呼び出すためのコードの例として、「オブジェクトプールインスタンス変数. setValue（伝票項目名, データ）」というコードが示されている。例えばプログラマは「オブジェクトプールインスタンス変数」部分及

び「伝票項目名」部分に項目データを特定するための値を設定し、且つ「データ」部分に格納すべきデータ（又はその変数）を設定することにより、適切なオブジェクトによる処理を規定することができる。伝票項目域 111 に対応するオブジェクトの項目更新部 522 は、このようなコードに従い、項目データを更新する。2 行目以降には、順に、DB 情報域 117、クラス間インターフェース領域 113、アプリケーション定数域 115 及びプロセッサ予約域 112 の各々について、その領域に対応するオブジェクトの項目更新部 522 を呼び出すためのコードの例が示されている。例えばプログラマは、このようなコードを業務クラスの雛型ソースプログラムに追加することにより、オブジェクトプールに格納されている項目データを更新する処理を規定することができる。

#### 【0053】

図 5 (c) には、共通インターフェースファイル群 11 の領域別に、図 4 に示したデータ状態値参照部 523 を呼び出すためのコードの例が示されている。まず 1 行目には、伝票項目域 111 に対応するオブジェクトのデータ状態値参照部 523 を呼び出すためのコードの例として、「オブジェクトプールインスタンス変数. getError (伝票項目名, 配列番号)」というコードが示されている。例えばプログラマは「オブジェクトプールインスタンス変数」部分、「伝票項目名」部分及び「配列番号」部分に項目データを特定するための値を設定することにより、適切なオブジェクトによる処理を規定することができる。伝票項目域 111 に対応するオブジェクトのデータ状態値参照部 523 は、このようなコードに従い、項目データの状態値を参照して値を返す。2 行目以降には、順に、DB 情報域 117、クラス間インターフェース領域 113、アプリケーション定数域 115 及びプロセッサ予約域 112 の各々について、その領域に対応するオブジェクトのデータ状態値参照部 523 を呼び出すためのコードの例が示されている。例えばプログラマは、このようなコードを業務クラスの雛型ソースプログラムに追加することにより、オブジェクトプールに格納されている項目データの状態値を参照する処理を規定することができる。

#### 【0054】

図 5 (d) には、共通インターフェースファイル群 11 の領域別に、図 4 に示したデータ状態値更新部 524 を呼び出すためのコードの例が示されている。まず 1 行目には、伝票項目域 111 に対応するオブジェクトのデータ状態値更新部 524 を呼び出すためのコードの例として、「オブジェクトプールインスタンス変数. setError (伝票項目名, データ)」というコードが示されている。例えばプログラマは「オブジェクトプールインスタンス変数」部分及び「伝票項目名」部分に項目データを特定するための値を設定し、且つ「データ」部分に格納すべき状態値（又はその変数）を設定することにより、適切なオブジェクトによる処理を規定することができる。伝票項目域 111 に対応するオブジェクトのデータ状態値更新部 524 は、このようなコードに従い、項目データの状態値を更新する。2 行目以降には、順に、DB 情報域 117、クラス間インターフェース領域 113、アプリケーション定数域 115 及びプロセッサ予約域 112 の各々について、その領域に対応するオブジェクトのデータ状態値更新部 524 を呼び出すためのコードの例が示されている。例えばプログラマは、このようなコードを業務クラスの雛型ソースプログラムに追加することにより、オブジェクトプールに格納されている項目データの状態値を更新する処理を規定することができる。

#### 【0055】

図 5 (e) には、共通インターフェースファイル群 11 の領域別に、図 4 に示したデータ格納数取得部 528 を呼び出すためのコードの例が示されている。まず 1 行目には、伝票項目域 111 に対応するオブジェクトのデータ格納数取得部 528 を呼び出すためのコードの例として、「オブジェクトプールインスタンス変数. getSize (伝票項目名)」というコードが示されている。例えばプログラマは「オブジェクトプールインスタンス変数」部分及び「伝票項目名」部分に項目を特定するための値を設定することにより、適切なオブジェクトによる処理を規定することができる。伝票項目域 111 に対応するオブジェクトのデータ格納数取得部 528 は、このようなコードに従い、データ格納数を取得して値を返す。2 行目以降には、順に、DB 情報域 117、クラス間インターフェース領域 1

13、アプリケーション定数域115及びプロセッサ予約域112の各々について、その領域に対応するオブジェクトのデータ格納数取得部528を呼び出すためのコードの例が示されている。例えばプログラマは、このようなコードを業務クラスの雛型ソースプログラムに追加することにより、オブジェクトプールに格納されている項目データ格納数を参照する処理を規定することができる。

#### 【0056】

図5(f)には、共通インターフェースファイル群11の領域別に、図4に示した項目初期化部525を呼び出すためのコードの例が示されている。まず1行目には、伝票項目域111に対応するオブジェクトの項目初期化部525を呼び出すためのコードの例として、「オブジェクトプールインスタンス変数. clearValue(伝票項目名)」というコードが示されている。例えばプログラマは「オブジェクトプールインスタンス変数」部分及び「伝票項目名」部分に項目を特定するための値を設定することにより、適切なオブジェクトによる処理を規定することができる。伝票項目域111に対応するオブジェクトの項目初期化部525は、このようなコードに従い、項目データの初期化を行う。2行目以降には、順に、DB情報域117、クラス間インターフェース領域113、アプリケーション定数域115及びプロセッサ予約域112の各々について、その領域に対応するオブジェクトの項目初期化部525を呼び出すためのコードの例が示されている。例えばプログラマは、このようなコードを業務クラスの雛型ソースプログラムに追加することにより、オブジェクトプールに格納されている項目データを、初期化する処理を規定することができる。

#### 【0057】

図5(g)には、伝票項目域111に対応するオブジェクトの属性参照部526(図4)を呼び出すためのコードの例として、「オブジェクトプールインスタンス変数. getAttributeVal(伝票項目名, 配列番号, 属性名)」というコードが示されている。例えばプログラマは「オブジェクトプールインスタンス変数」部分、「伝票項目名」部分、「配列番号」部分及び「属性名」部分に属性データを特定するための値を設定することにより、適切なオブジェクトによる処理を規定することができる。伝票項目域111に対応するオブジェクトの属性参照部526は、このようなコードに従い、属性データを参照して値を返す。例えばプログラマは、このようなコードを業務クラスの雛型ソースプログラムに追加することにより、オブジェクトプールに格納されている項目データの属性値を、参照する処理をコーディングすることができる。

#### 【0058】

図5(h)には、伝票項目域111に対応するオブジェクトの属性更新部527(図4)を呼び出すためのコードの例として、「オブジェクトプールインスタンス変数. setAttributeVal(伝票項目名, 配列番号, 属性名, 属性値)」というコードが示されている。例えばプログラマは「オブジェクトプールインスタンス変数」部分、「伝票項目名」部分、「配列番号」部分及び「属性名」部分に属性データを特定するための値を設定し、且つ「属性値」部分に格納すべき属性値(又はその変数)を設定することにより、適切なオブジェクトによる処理を規定することができる。伝票項目域111に対応するオブジェクトの属性更新部527は、このようなコードに従い、属性データの更新を行う。例えばプログラマは、このようなコードを業務クラスの雛型ソースプログラムに追加することにより、オブジェクトプールに格納されている項目データの属性値を更新する処理を規定することができる。

#### 【0059】

### 2. 業務クラス実行時の処理

業務クラスを実行する際における、本発明の一実施の形態に係るコンピュータ・システムの機能ブロック図を図6に示す。コンピュータ・ネットワーク201には、1又は複数の端末装置203と、本実施の形態における主たる処理を実施するサーバ205とが接続されている。端末装置203は、XML対応のブラウザを有している。また、サーバ205は、端末装置203のブラウザとのインターフェースとなるサブレット2051と、



入力画面用XMLデータ2054や出力画面用XMLデータ2055を格納するメモリ領域2060と、入力画面のための雛型XMLデータ2062とXSLデータ2063とスキーマ2064とを格納する第1XMLデータ格納部2061と、XMLデータを解析して対応する業務クラスを呼び出す処理などを実施するプロセッサ2071と、業務クラス管理ファイル15と、業務クラス群がロードされるメモリ領域2080と、出力画面のための雛型XMLデータ2092とXSLデータ2093とスキーマ2094とを格納する第2XMLデータ格納部2091と、共通インターフェースファイルがロードされ且つインデックス情報2102を保持するメモリ領域2101（オブジェクトプール）と、実行可能な業務クラスのファイルが格納された業務クラス格納部2111と、オブジェクトプールにおけるオブジェクトを構成するためのクラスファイルであって且つ実行可能な共通インターフェースファイルが格納された共通インターフェースファイル格納部2112と、初期設定ファイル119とが含まれる。

#### 【0060】

サーバ2051には、端末装置203のブラウザからのリクエストに応答するための処理を実施するリクエスト応答部2052と、端末装置203のブラウザから受信した選択又は入力データに対応してXMLデータ（入力画面用XMLデータ2054）を生成及び出力するXML生成部2056と、プロセッサ2071から処理結果を含むXMLデータ（出力画面用XMLデータ2055）を受信し、端末装置203のブラウザに出力する結果出力部2053とが含まれる。

#### 【0061】

プロセッサ2071には、メモリ領域2101を初期化（すなわち、共通インターフェースファイルを共通インターフェースファイル格納部2112からロードし且つインデックス情報2102を生成）したり、業務クラスを業務クラス格納部2111からメモリ領域2080にロードするなどの初期処理を実施する初期処理部2072と、入力画面用XMLデータ2054を処理するXMLデータ処理部2073と、メモリ領域2101の伝票項目域2103に格納されたデータ等と第2XMLデータ格納部2091に格納された雛型XMLデータ2092とに基づき出力画面用XMLデータ2055を生成する出力XML生成部2074とが含まれる。なお、メモリ領域2101のDB情報域2107は例えばハッシュ・テーブル形式でデータを格納するようになっており、インデックス情報2102にデータの格納場所に関するデータが格納される。

#### 【0062】

次に図7乃至図13を用いて図6に示したコンピュータ・システム200の処理フローを説明する。まず、サーバ205におけるサーバ2051のリクエスト応答部2052は、端末装置203のブラウザから特定の入力画面データの要求を受信する（ステップS31）。そうすると、リクエスト応答部2052は、特定の入力画面用の雛型XMLデータ2062及び当該雛型XMLデータ2062に対応するXSLデータ2063を第1XMLデータ格納部2061から読み出し、端末装置203のブラウザに送信する（ステップS33）。なお、実際の処理では、リクエスト応答部2052は、まず雛型XMLデータ2062を端末装置203のブラウザに送信しておき、端末装置203のブラウザからの雛型XMLデータ2062に対応するXSLデータの送信要求を受信することにより、XSLデータ2063を端末装置203のブラウザに送信するが、本実施の形態の主旨ではないため、上の説明及び以下の同様の処理についての説明では、XMLデータとXSLデータとが併せて送信されるように記載を簡略化している。

#### 【0063】

端末装置203のブラウザは、サーバ205から特定の入力画面用の雛型XMLデータ2062及び雛型XMLデータ2062に対応するXSLデータ2063を受信すると、受信したデータに基づきHTMLデータを生成し、表示装置に表示する。例えば、図8に示すような画面が端末装置203の表示装置に表示される。図8の例は、入金取引の入力画面であって、銀行名の入力又は選択欄601と、支店名の入力又は選択欄602と、口座種別の選択欄（ラジオボタン）603と、口座番号の入力欄604と、取引金額の入力

欄605と、確認ボタン606と、戻るボタン607とが含まれる。すなわち、伝票名及び伝票項目に対応するタグが、雛型XMLデータ2062には含まれている。

#### 【0064】

端末装置203のユーザは、端末装置203を操作してデータを選択又は入力して確認ボタン606をクリックする。そうすると、端末装置203のブラウザは、例えばHTTP (Hyper Text Transfer Protocol) のPOSTメソッドにより入力又は選択データをサーバ205に送信する。サーバ205におけるサブレット2051のXML生成部2056は、端末装置203のブラウザから入力又は選択データを受信し、一旦メモリに格納する(ステップS35)。そして、XML生成部2056は、入力又は選択データを解析し、入力画面用XMLデータ2054を生成し、メモリ領域2060に格納する(ステップS37)。なお、XML生成部2056は、例えばXMLデータ2063を参照して入力又は選択データを分節し、雛型XMLデータ2062に埋め込むことにより、入力画面用XMLデータ2054を生成する。また、スキーマ2064を用いて入力画面用XMLデータ2054の妥当性を確認する。そして、XML生成部2056は、プロセッサ2071を呼び出し、入力画面用XMLデータ2054をプロセッサ2071に出力する(ステップS39)。

#### 【0065】

入力画面用XMLデータ2054の一例を図9に示す。図9の例では、伝票名に対応する入金取引入力タグと、200という実データを挟持する銀行名タグ、121という実データを挟持する支店名タグ、01という実データを挟持する口座種別タグ、12345678という実データを挟持する口座番号タグと、10000という実データを挟持する取引金額タグとが含まれる。

#### 【0066】

一方、サーバ205におけるプロセッサ2071の初期処理部2072は、例えば端末装置203とのセッションが張られた時に、オブジェクトプールの初期化を実施する(ステップS41)。例えば業務クラス管理ファイル15を参照して、オブジェクトプールとして列挙されたクラスに係る共通インターフェースファイルを共通インターフェースファイル格納部2112からメモリ領域2101にロードする。なお、共通インターフェースファイル格納部2112には、共通インターフェースファイル群11の各領域に格納された共通インターフェース雛型ソースファイルに基づき作成された、実行可能な共通インターフェースファイルが格納されている。

#### 【0067】

また、初期処理部2072は、業務クラス管理ファイル15を参照して、業務クラス格納部2111に格納された全ての業務クラスのファイルをメモリ領域2080にロードする(ステップS43)。そして、例えばプロセス初期処理クラスを呼び出し、実行させる(ステップS45)。プロセス初期処理クラスは、図3のステップS5において生成された雛型ソースプログラムを基に適切にコーディングされたクラスである。また、伝票項目域2103等を初期化する(ステップS47)。この際、アプリケーション定数域2106には、初期設定ファイル119に基づく項目データがプロセス初期処理クラスにより格納される。

#### 【0068】

そして、ステップS39でサブレット2051が入力画面用XMLデータ2054を出力したことに応じて、プロセッサ2071のXMLデータ処理部2073は、入力画面用XMLデータ2054を受信する(ステップS49)。プロセッサ2071のXMLデータ処理部2073は、入力画面用XMLデータ2054を解析及び文節して、実データ(入力又は選択データ)を伝票項目域2103の対応する領域(オブジェクト)に格納する(ステップS51)。図9のような入力画面用XMLデータ2054を処理する場合には、銀行名のデータ格納領域、支店名のデータ格納領域、口座種別のデータ格納領域、口座番号のデータ格納領域、及び取引金額のデータ格納領域に、実データを格納する。

#### 【0069】

プロセッサ2071のXMLデータ処理部2073は、入力画面用XMLデータ2054の伝票名に対応する前処理クラス（ここでは入金取引入力前処理クラス2081）を呼び出し、業務処理を実行させる（ステップS53）。また、終了タグ以外のタグ毎に（ステップS55：Noルート）、業務クラス（銀行名クラス2082、支店名クラス2083、口座種別クラス2084、口座番号クラス2085、取引金額クラス2086）をそれぞれ呼び出し、業務処理を実行させる（ステップS57）。そして、入力画面XMLデータ2054の終了タグが検出されると（ステップS55：Yesルート）、伝票名に対応する伝票処理クラス（入金取引入力伝票処理クラス2087）を呼び出し、業務処理を実施させる（ステップS59）。そして、伝票名に対応する後処理クラス（入力取引入力後処理クラス2088）を呼び出し、実行させる（ステップS61）。

#### 【0070】

これら呼び出された業務クラスは、メモリ領域2101（オブジェクトプール）の各領域に属するオブジェクトに格納されているデータを参照又は更新する等、適宜利用して処理を実行する。なお、業務クラスにおける処理において、手数料、合計金額、残高、取扱店番、取扱支店名、取引日、時刻といったデータも別途生成されて、例えば伝票項目域2103に属するオブジェクト及びDB情報域2107に属するオブジェクトに格納される。

#### 【0071】

図10A乃至図10Dに、オブジェクトプールを用いた処理の概念図を示す。なお、図10A乃至図10Dには、処理の概要を説明するために必要な構成要素のみを示しており、その他、実際の処理上必要となる構成要素を省略している場合がある。図10Aには、伝票項目域2103に属するオブジェクトを用いた処理の例が示されている。図10Aの例では、例えば端末装置203の表示装置に表示されているブラウザ画面1001には、入力項目1002及び入力項目1003が含まれている。そこでユーザが端末装置203を操作して入力項目1002及び入力項目1003の少なくともいずれかにデータを入力し、図示しない確認ボタン等をクリックする。そうすると入力項目1002及び1003に対応する入力データはサーバ205に送信され、サーバレット2051によりXMLデータ1004（図6における入力画面用XMLデータ2054に相当）が生成される。XMLデータ1004は、例えばXML文1005に示したような内容を含んでいる。すなわち、入力項目の属性1の値及び属性2の値と、ユーザから入力された実データとが含まれている。プロセッサ2071は、XMLデータ1004のタグ名に基づき、オブジェクトプール（ここでは伝票項目域2103）の中から入力項目に対応するオブジェクトを特定し、属性1の値及び属性2の値とユーザから入力された実データとを当該オブジェクトに出力する。そして、例えばオブジェクト1008はデータ領域1010に実データを格納し、属性領域1011に属性1の値及び属性2の値を格納する。図10Aに示した例のように属性が複数定義されている場合には、複数の属性の値が属性領域1011に格納される。なお、プロセッサ2071はXMLデータ1004のタグに基づき業務クラス1012をコールする。業務クラス1012は、必要に応じてオブジェクトプール2103内のオブジェクト（例えばオブジェクト1008）を呼び出し、データ領域1010と属性領域1011とに格納されたデータの参照処理や更新処理を行うことにより、伝票処理を実施する。

#### 【0072】

図10Bには、DB情報域2107に属するオブジェクトを用いた処理の例が示されている。例えばプロセッサ2071からコールされた業務クラス1013は、オブジェクトプール（ここではDB情報域2107）のオブジェクト1021に対して、DB1024から抽出したレコードのデータを出力する。そして、オブジェクト1021は、データ領域1023にデータを格納する。業務クラス1013がDB1024から複数のレコードを抽出していた場合には、当該複数のレコードのデータを各々データ領域1023に格納する。なお、データ領域1023は、例えばハッシュ・テーブルである。

#### 【0073】

図10Cには、アプリケーション定数域2106に属するオブジェクトを用いた処理の例が示されている。上でも述べたが、プロセッサ2071（初期処理部2072）は、オブジェクトプールの初期化を実施する際、プロセス初期処理クラス1014を呼び出す。プロセス初期処理クラス1014は、初期設定ファイル119を読み込み、アプリケーション定数項目データをアプリケーション定数域2106のオブジェクトに格納する。例えばオブジェクト1031及びオブジェクト1032にデータが格納される。そして例えば業務クラス1015は、必要に応じてオブジェクト1031やオブジェクト1032を呼び出し、格納されているデータを参照することにより、伝票処理を実施する。

#### 【0074】

図10Dには、クラス間インターフェース領域2105に属するオブジェクトを用いた処理の例が示されている。図10Dでは、業務クラスであるクラスA（1045）からクラスB（1046）にデータを渡し、クラスB（1046）からクラスA（1045）にデータを返すという処理を行う場合の例が示されている。まず、クラスA（1045）は、矢印1047に示すように、オブジェクトプール（ここではクラス間インターフェース領域2105）内のクラスBのインターフェース項目域に属するオブジェクト1041にデータを出力する。そしてオブジェクト1041はデータを入力領域1042に格納する。次にクラスB（1046）は、矢印1049に示すように、入力領域1042に格納されたデータを参照する。そしてクラスB（1046）は、矢印1050に示すように、オブジェクト1041に対し、参照データに基づき所定の処理を行うことにより生成されたデータを出力領域1043に格納する。そして、クラスA（1045）は、矢印1048に示すように、出力領域1043に格納されたデータを参照する。このように、業務クラス間では直接データのやり取りを行うことなく、クラス間インターフェース領域2105に属するオブジェクトを介してデータのやり取りを行う。これにより、各業務クラスを他の業務クラスを考慮することなく作成することができるため、プログラムの作成効率が向上する。

#### 【0075】

なお、各業務クラスの具体的な処理内容については、業務ロジックに即した形でコーディングされ、上でも述べたが、特定の伝票の処理は、入力画面用XMLデータ2054に基づきXMLデータ処理部2073により呼び出される業務クラスによる処理で過不足無く完結するように構成されている。また、XMLデータ処理部2073は、入力画面用XMLデータ2054に含まれるタグに、実際に存在しないような業務クラスに対応するタグが含まれる場合には、業務クラスの呼び出しに失敗する。そのような場合には、適切な処理が実施できない又は不適切な入力が発生しているので、エラーをサブレット2051に出力して、処理を終了させる。

#### 【0076】

全ての処理が完了すると、後処理クラス（入金取引入力後処理クラス2088）は、第2XMLデータの生成要求（すなわち、出力画面用XMLの文書名を含む出力伝票生成要求）を、プロセッサ2071の出力XML生成部2074に出力する。処理は、図7の端子Aを介して図11の処理に遷移する。

#### 【0077】

出力XML生成部2074は、出力画面用XMLデータの生成要求に応じて、第2XMLデータ格納部2091から該当する雛型XMLデータ2092を取得する（ステップS63）。この際、出力画面用XMLデータに対応するスキーマ2094も併せて取得する。また、メモリ領域2101の伝票項目域2103に属するオブジェクトから、読み出した雛型XMLデータ2092に含まれるタグに対応するデータ項目の実データを受け取り、雛型XMLデータ2092に埋め込むことにより、出力画面用XMLデータ2055を生成し、メモリ領域2060に格納する。この際、スキーマ2094を用いて出力画面用XMLデータ2055の妥当性を確認する。そして、当該出力画面用XMLデータ2055をサブレット2051に出力する（ステップS65）。

**【0078】**

出力画面用XMLデータ2055の一例を図12に示す。図12の例では、スタイルシート（出力画面、xs1）を指定するタグと、伝票名を表すタグ（入金取引結果タグ）と、伝票項目を表すタグ（取引日タグ、時刻タグ、銀行名タグ、支店名タグ、口座種別タグ、口座番号タグ、取引金額タグ、手数料タグ、合計金額タグ、残高タグ、取扱店番タグ、取扱支店名タグ）と、それぞれのタグに対応する実データとが含まれる。

**【0079】**

なお、プロセッサ2071側でプロセスが終了とは判断されなければ端子Bを介して図7のステップS47に戻る。一方、プロセスが終了と判断されれば、プロセス終了クラスを呼び出し、実行させる（ステップS73）。プロセス終了クラスは、図3のステップS5において生成された雛型ソースプログラムを基に、適切にコーディングされたクラスである。

**【0080】**

サーバレット2051の結果出力部2053は、プロセッサ2071から出力画面用XMLデータ2055を受け取り（ステップS67）、当該出力画面用XMLデータ2055及び第2XMLデータ格納部2091から読み出されたXSLデータ2093を端末装置203に出力する（ステップS69）。端末装置203のブラウザは、サーバ205から出力画面用XMLデータ2055及びXSLデータ2093を受信し、HTMLデータを生成して表示装置に表示する。例えば図13のような画面を表示する。図13の例では、出力伝票名（入金取引（結果））、取引日、時刻、銀行名、支店名、口座種別、口座番号、取引金額、手数料、合計金額、残高、取扱店番、取扱支店名の各表示欄が設けられている。さらに、確認ボタン1301とメニューに戻る初画面ボタン1302も設けられる。

**【0081】**

このように、業務クラスを適切に呼び出し、処理を実施させることにより、伝票を含む帳票の処理が適切に行われる。なお、サーバレット2051及びプロセッサ2071については全システム共通で用いることができ、業務（伝票）ごとに異なる部分については業務クラスを作成する。このようなプログラム開発手法を採用することにより、プログラムの再利用、開発の分業及び開発期間の短縮を図ることができる。さらに、業務クラスを機能を限定して他の業務クラスから分離して作成するので、プログラムの品質向上に繋がる。

**【0082】**

以上本発明の一実施の形態を説明したが、本発明はこれに限定されるものではない。例えば、図1及び図6に示した機能ブロック図は、本実施の形態を説明する上で分離した機能であり、必ずしもプログラムのモジュールに対応するものではない。

**【0083】**

（付記1）

XMLデータ格納部に格納され且つ帳票画面に対応するXMLデータを解析し、前記XMLデータに含まれるタグに対応する業務クラスを特定する業務クラス特定ステップと、生成すべき業務クラスの項目が登録された業務クラス管理部を参照して、特定された前記業務クラスの項目が未登録であるか判断するステップと、

特定された前記業務クラスの項目が未登録である場合には、前記業務クラス管理部に当該特定された前記業務クラスの項目を登録するステップと、  
を含み、コンピュータにより実行される情報処理方法。

**【0084】**

（付記2）

特定された前記業務クラスの項目が未登録である場合には、当該特定された前記業務クラスのための雛型ソースプログラム・データを生成し、ソースプログラム格納部に格納するステップ

をさらに含む付記1記載の情報処理方法。

## 【 0 0 8 5 】

## (付記 3)

H T M L ファイル格納部から帳票画面のための H T M L ファイルを読み出し、所定のルールに従って前記帳票画面に対応する X M L データを生成し、前記 X M L データ格納部に格納するステップ

をさらに含む付記 1 記載の情報処理方法。

## 【 0 0 8 6 】

## (付記 4)

前記業務クラス特定ステップが、

前記帳票画面に対応する X M L データの開始又は終了タグに対応して、前処理クラス、後処理クラス及び帳票処理クラスを特定するステップ

を含む付記 1 記載の情報処理方法。

## 【 0 0 8 7 】

## (付記 5)

ユーザの指示に応じて帳票画面に対応する H T M L ファイルを生成し、前記 H T M L ファイル格納部に格納するステップ

をさらに含む付記 3 記載の情報処理方法。

## 【 0 0 8 8 】

## (付記 6)

前記 X M L データ格納部に格納され且つ帳票画面に対応する X M L データに含まれるタグにより帳票項目を特定するステップと、

生成すべき帳票項目格納オブジェクトが登録された帳票項目格納オブジェクト管理部を参照して、特定された前記帳票項目が未登録であるか判断するステップと、

特定された前記帳票項目が未登録である場合には、前記帳票項目格納オブジェクト管理部に当該特定された前記帳票項目を登録するステップと、

をさらに含む付記 1 記載の情報処理方法。

## 【 0 0 8 9 】

## (付記 7)

X M L データ格納部に格納され且つ帳票画面に対応する X M L データを解析し、前記 X M L データに含まれるタグに対応する業務クラスを特定する業務クラス特定ステップと、

生成すべき業務クラスの項目が登録された業務クラス管理部を参照して、特定された前記業務クラスの項目が未登録であるか判断するステップと、

特定された前記業務クラスの項目が未登録である場合には、当該特定された前記業務クラスのための雛型ソースプログラム・データを生成し、ソースプログラム格納部に格納するステップと、

を含み、コンピュータにより実行される情報処理方法。

## 【 0 0 9 0 】

## (付記 8)

帳票画面に対する入力又は選択データに対応するタグを含む X M L データをメモリに格納し、当該 X M L データに含まれるタグに対応し且つ当該タグに関連する処理を実施するためのプログラムである業務クラスを特定する業務クラス特定ステップと、

予め定義され且つメモリにロードされた業務クラスの中で、特定された前記業務クラスを呼び出す呼出ステップと、

を含み、コンピュータにより実行される情報処理方法。

## 【 0 0 9 1 】

## (付記 9)

前記業務クラス特定ステップが、

帳票画面に係る帳票と前記 X M L データに含まれるタグとに対応する前処理クラス、後処理クラス及び帳票処理クラスを特定するステップ

を含む付記 8 記載の情報処理方法。

## 【 0 0 9 2 】

## (付記 1 0)

前記帳票画面に対する入力又は選択データを、予め定義され且つメモリにロードされた帳票項目格納オブジェクトに出力し、当該帳票項目格納オブジェクトにより前記帳票画面に対する入力又は選択データをメモリに格納するステップと、

呼び出された前記業務クラスと前記帳票項目格納オブジェクトとがデータをやり取りすることにより処理を実施する処理実施ステップと、

をさらに含む付記 8 記載の情報処理方法。

## 【 0 0 9 3 】

## (付記 1 1)

前記処理実施ステップが、

第 1 の業務クラスから第 2 の業務クラスへデータを渡す場合に、前記第 1 の業務クラスが、予め定義され且つメモリにロードされたクラス間インターフェース・オブジェクトにデータを出力するステップと、

前記第 2 の業務クラスが前記クラス間インターフェース・オブジェクトを参照して、前記データを受け取るステップと、

を含む付記 1 0 記載の情報処理方法。

## 【 0 0 9 4 】

## (付記 1 2)

帳票画面に対する入力又は選択データを帳票画面を表示した装置から受信し、当該帳票画面に対する入力又は選択データ及び対応するタグを含む XML データを生成するステップ

をさらに含む付記 8 記載の情報処理方法。

## 【 0 0 9 5 】

## (付記 1 3)

呼び出された前記業務クラスにより出力要求を受けた場合、前記帳票項目格納オブジェクトに保持されたデータを用いて出力用 XML データを生成し、メモリに格納するステップと、

前記メモリに格納された前記出力用 XML データを前記帳票画面を表示した装置に出力するステップと、

をさらに含む付記 1 0 記載の情報処理方法。

## 【 0 0 9 6 】

## (付記 1 4)

特定された前記業務クラスが存在しない場合、エラー情報を生成し、出力するステップをさらに含む付記 8 記載の情報処理方法。

## 【 0 0 9 7 】

## (付記 1 5)

前記帳票画面に係る帳票に対応して当該帳票全体の処理シーケンスが規定されておらず、呼び出された前記業務クラスの処理にて当該帳票に対する処理が完了するように、前記業務クラスの各々が構成されていることを特徴とする付記 8 記載の情報処理方法。

## 【 0 0 9 8 】

## (付記 1 6)

XML データ格納部に格納され且つ帳票画面に対応する XML データを解析し、前記 XML データに含まれるタグに対応する業務クラスを特定する業務クラス特定ステップと、

生成すべき業務クラスの項目が登録された業務クラス管理部を参照して、特定された前記業務クラスの項目が未登録であるか判断するステップと、

特定された前記業務クラスの項目が未登録である場合には、前記業務クラス管理部に当該特定された前記業務クラスの項目を登録するステップと、

をコンピュータに実行させるプログラム。

## 【 0 0 9 9 】

(付記 17)

XML データ格納部に格納され且つ帳票画面に対応する XML データを解析し、前記 XML データに含まれるタグに対応する業務クラスを特定する業務クラス特定ステップと、  
生成すべき業務クラスの項目が登録された業務クラス管理部を参照して、特定された前記業務クラスの項目が未登録であるか判断するステップと、

特定された前記業務クラスの項目が未登録である場合には、当該特定された前記業務クラスのための雛型ソースプログラム・データを生成し、ソースプログラム格納部に格納するステップと、

をコンピュータに実行させるプログラム。

【0100】

(付記 18)

帳票画面に対する入力又は選択データに対応するタグを含む XML データをメモリに格納し、当該 XML データに含まれるタグに対応し且つ当該タグに関連する処理を実施するためのプログラムである業務クラスを特定する業務クラス特定ステップと、

予め定義され且つメモリにロードされた業務クラスの中で、特定された前記業務クラスを呼び出す呼出ステップと、

をコンピュータに実行させるプログラム。

【0101】

(付記 19)

XML データ格納部に格納され且つ帳票画面に対応する XML データを解析し、前記 XML データに含まれるタグに対応する業務クラスを特定する手段と、

生成すべき業務クラスの項目が登録された業務クラス管理部を参照して、特定された前記業務クラスの項目が未登録であるか判断する手段と、

特定された前記業務クラスの項目が未登録である場合には、前記業務クラス管理部に当該特定された前記業務クラスの項目を登録する手段と、

を有するコンピュータ・システム。

【0102】

(付記 20)

XML データ格納部に格納され且つ帳票画面に対応する XML データを解析し、前記 XML データに含まれるタグに対応する業務クラスを特定する手段と、

生成すべき業務クラスの項目が登録された業務クラス管理部を参照して、特定された前記業務クラスの項目が未登録であるか判断する手段と、

特定された前記業務クラスの項目が未登録である場合には、当該特定された前記業務クラスのための雛型ソースプログラム・データを生成し、ソースプログラム格納部に格納する手段と、

を有するコンピュータ・システム。

【0103】

(付記 21)

帳票画面に対する入力又は選択データに対応するタグを含む XML データをメモリに格納し、当該 XML データに含まれるタグに対応し且つ当該タグに関連する処理を実施するためのプログラムである業務クラスを特定する手段と、

予め定義され且つメモリにロードされた業務クラスの中で、特定された前記業務クラスを呼び出す手段と、

を有するコンピュータ・システム。

【図面の簡単な説明】

【0104】

【図 1】 業務クラス作成時における機能ブロック図である。

【図 2】 業務クラス管理ファイルに格納されるデータの一例を示す図である。

【図 3】 業務クラス作成時における処理フローを示す図である。

【図 4】 オブジェクトの模式図である。



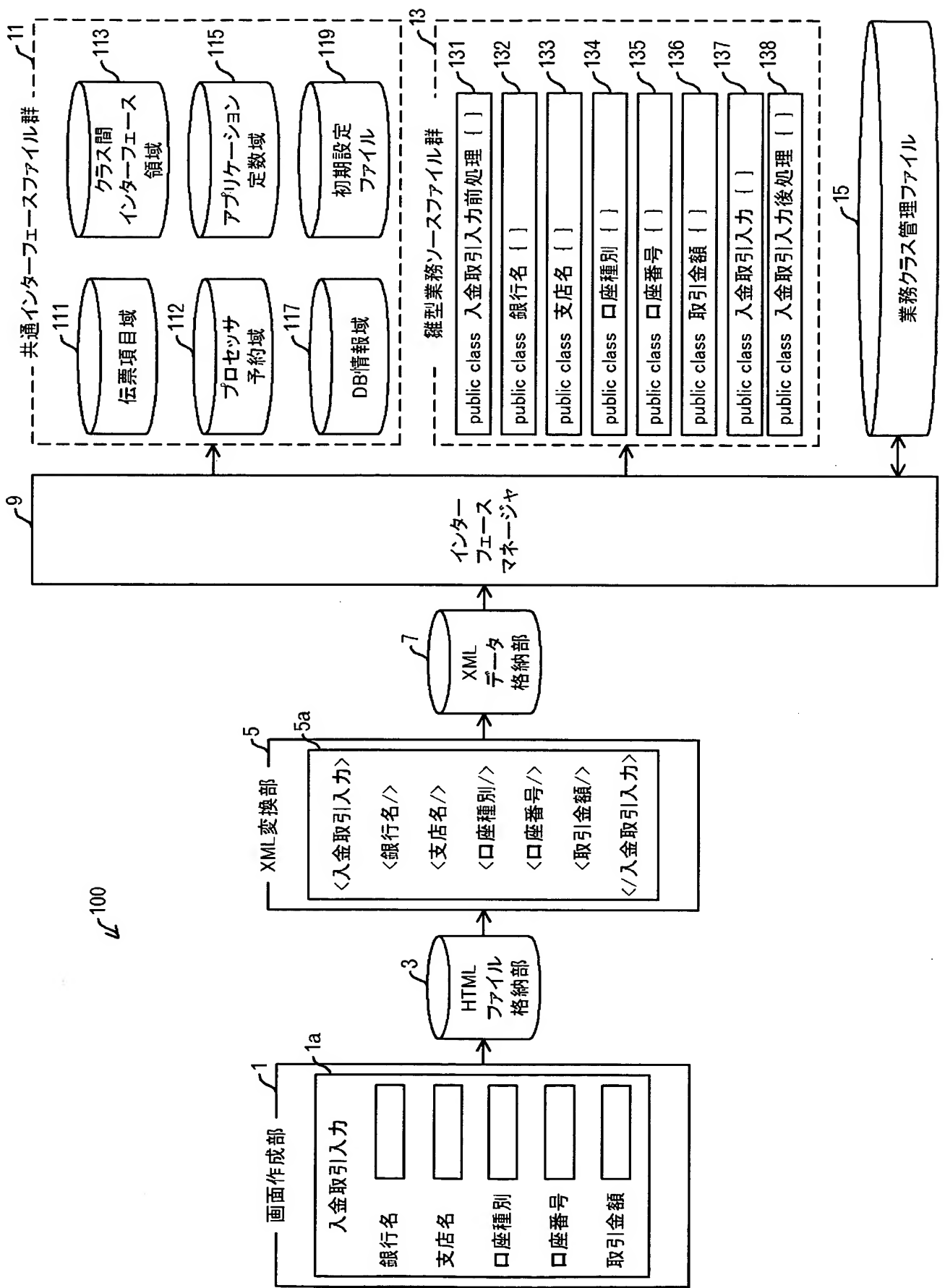
- 【図5】 オブジェクトを使用するためのコードの例を示す図である。  
【図6】 業務クラス実行時における機能ブロック図である。  
【図7】 業務クラス実行時における第1の処理フローを示す図である。  
【図8】 入力画面例を示す図である。  
【図9】 入力画面用XMLデータの一例を示す図である。  
【図10A】 オブジェクトプールを用いた処理の第1の例を示す図である。  
【図10B】 オブジェクトプールを用いた処理の第2の例を示す図である。  
【図10C】 オブジェクトプールを用いた処理の第3の例を示す図である。  
【図10D】 オブジェクトプールを用いた処理の第4の例を示す図である。  
【図11】 業務クラス実行時における第2の処理フローを示す図である。  
【図12】 出力画面用XMLデータの一例を示す図である。  
【図13】 出力画面例を示す図である。

【符号の説明】

【0105】

- 1 画面作成部      3 HTMLファイル格納部      5 XML変換部  
7 XMLデータ格納部      9 インターフェースマネージャ  
11 共通インターフェースファイル群      13 雛型業務ソースファイル群  
15 業務クラス管理ファイル  
201 コンピュータ・ネットワーク      203 端末装置      205 サーバ  
2051 サブレット      2061 第1XMLデータ格納部  
2071 プロセッサ      2111 業務クラス格納部  
2091 第2XMLデータ格納部  
2060, 2080, 2101 メモリ領域

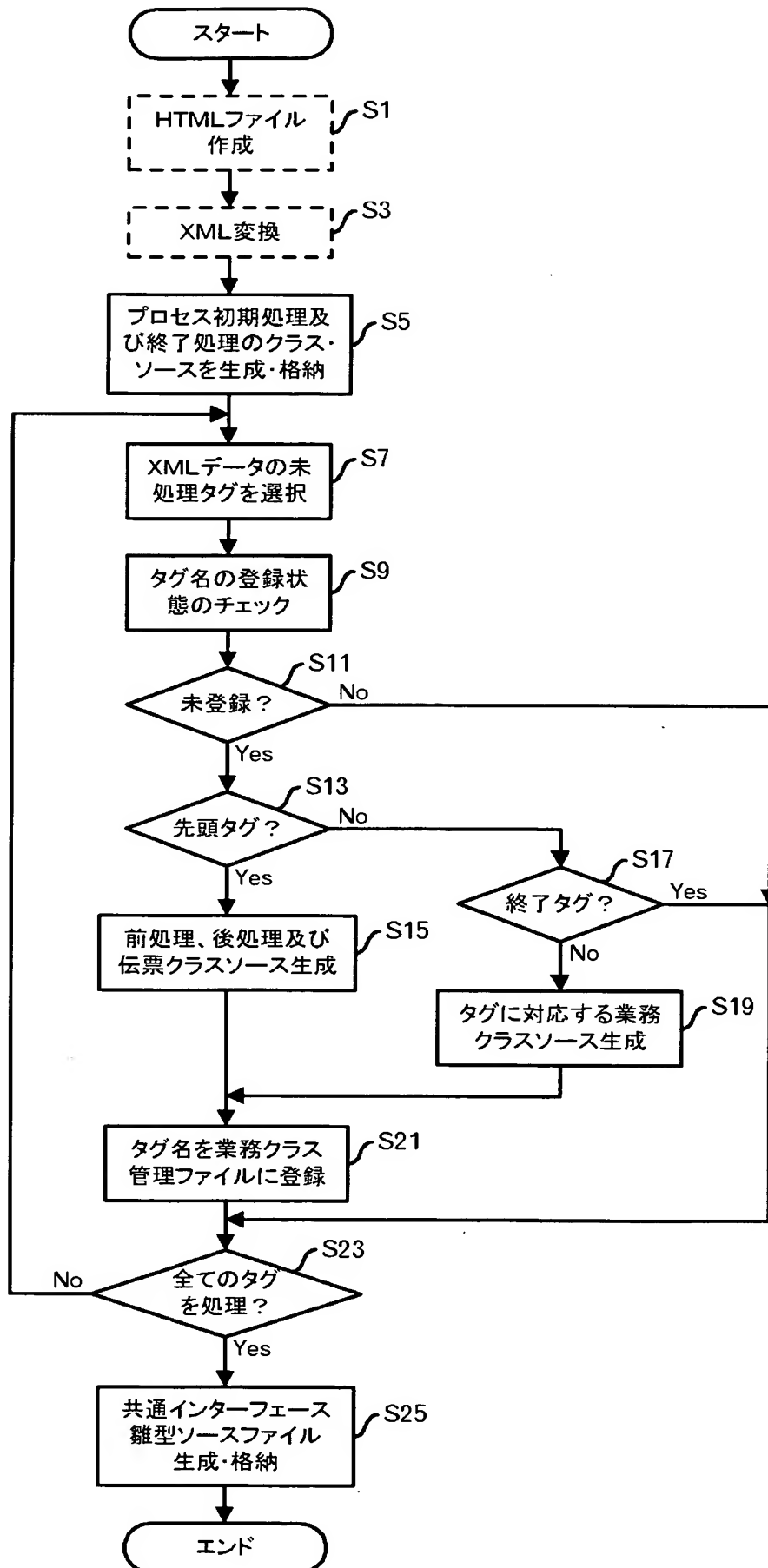
【書類名】 図面  
【図 1】



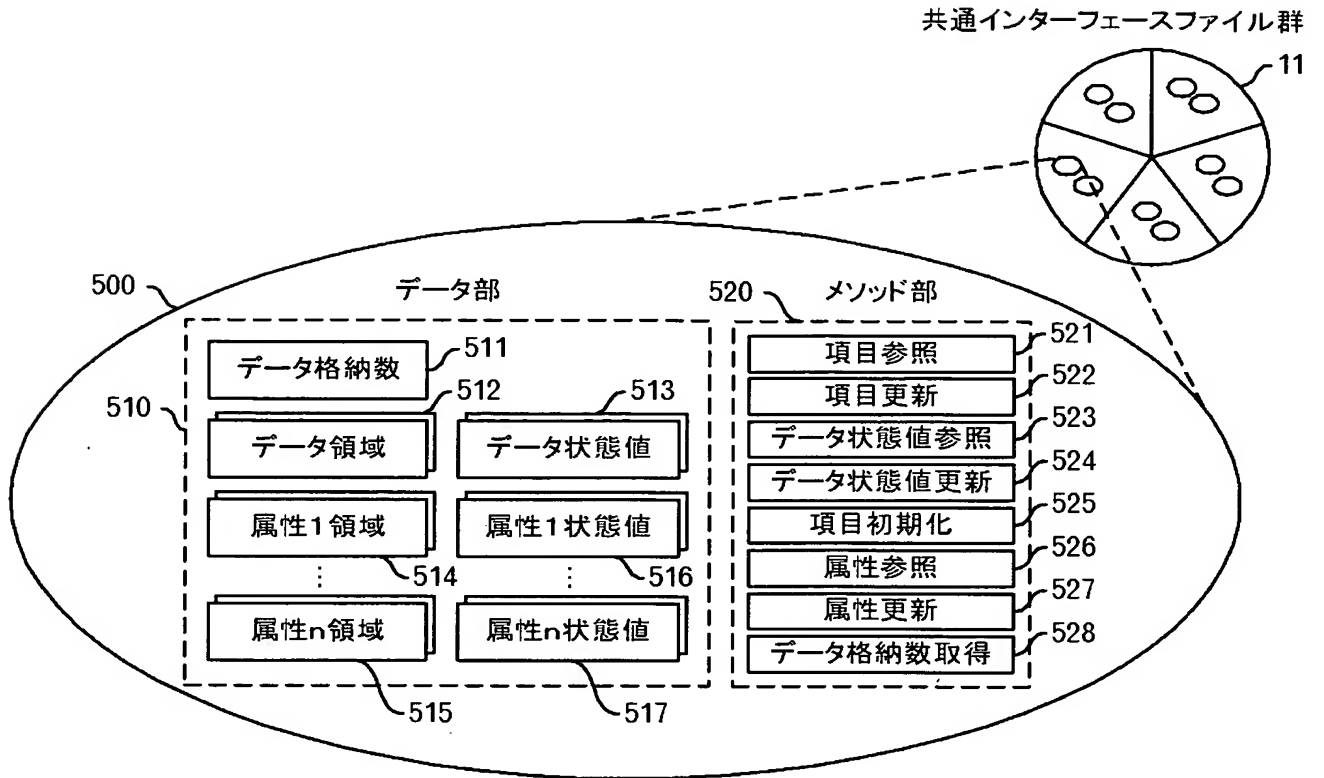
## 【図 2】

```
<?xml version="1.0" encoding="Shift_JIS"?>
- <資産管理>
- <オブジェクトプール>
+ <プロセッサ予約域>
- <伝票項目域>
  <入金取引入力 ルート判定="ルート" 登録日時="2003/7/23 10:54:40" />
  <取引金額 登録日時="2003/7/23 10:54:40" />
  <口座番号 登録日時="2003/7/23 10:54:40" />
  <口座種別 登録日時="2003/7/23 10:54:40" />
  <支店名 登録日時="2003/7/23 10:54:40" />
  <銀行名 登録日時="2003/7/23 10:54:40" />
  </伝票項目域>
  <DB情報域 />
  <アプリ定数項目域 />
+ <クラス間インタフェース項目域>
</オブジェクトプール>
- <雛形業務クラス>
  <USR_取引金額 登録日時="2003/7/23 10:54:40" />
  <USR_口座番号 登録日時="2003/7/23 10:54:40" />
  <USR_口座種別 登録日時="2003/7/23 10:54:40" />
  <USR_支店名 登録日時="2003/7/23 10:54:40" />
  <USR_銀行名 登録日時="2003/7/23 10:54:40" />
  <USR_入金取引入力前処理 登録日時="2003/7/23 10:54:40" />
  <USR_入金取引入力後処理 登録日時="2003/7/23 10:54:40" />
  <USR_入金取引入力 登録日時="2003/7/23 10:54:40" />
</雛形業務クラス>
</資産管理>
```

【図 3】



【図 4】



【図 5】

(a)

オブジェクトプールインスタンス変数. getValue(伝票項目名, 配列番号)
オブジェクトプールインスタンス変数. getDBValue(DB項目名, 配列番号)
オブジェクトプールインスタンス変数. getAphandleValue(クラス間インターフェース項目名, 配列番号)
オブジェクトプールインスタンス変数. getApconstValue(アプリケーション定数項目名, 配列番号)
オブジェクトプールインスタンス変数. getProcessorValue(プロセッサ予約項目名, 配列番号)

(b)

オブジェクトプールインスタンス変数. setValue(伝票項目名, データ)
オブジェクトプールインスタンス変数. setDBValue(DB項目名, データ)
オブジェクトプールインスタンス変数. setAphandleValue(クラス間インターフェース項目名, データ)
オブジェクトプールインスタンス変数. setApconstValue(アプリケーション定数項目名, データ)
オブジェクトプールインスタンス変数. setProcessorValue(プロセッサ予約項目名, データ)

(c)

オブジェクトプールインスタンス変数. getError(伝票項目名, 配列番号)
オブジェクトプールインスタンス変数. getDBError(DB項目名, 配列番号)
オブジェクトプールインスタンス変数. getAphandleError(クラス間インターフェース項目名, 配列番号)
オブジェクトプールインスタンス変数. getApconstError(アプリケーション定数項目名, 配列番号)
オブジェクトプールインスタンス変数. getProcessorError(プロセッサ予約項目名, 配列番号)

(d)

オブジェクトプールインスタンス変数. setError(伝票項目名, データ)
オブジェクトプールインスタンス変数. setDBError(DB項目名, データ)
オブジェクトプールインスタンス変数. setAphandleError(クラス間インターフェース項目名, データ)
オブジェクトプールインスタンス変数. setApconstError(アプリケーション定数項目名, データ)
オブジェクトプールインスタンス変数. setProcessorError(プロセッサ予約項目名, データ)

(e)

オブジェクトプールインスタンス変数. getSize(伝票項目名)
オブジェクトプールインスタンス変数. getDBSize(DB項目名)
オブジェクトプールインスタンス変数. getAphandleSize(クラス間インターフェース項目名)
オブジェクトプールインスタンス変数. getApconstSize(アプリケーション定数項目名)
オブジェクトプールインスタンス変数. getProcessorSize(プロセッサ予約項目名)

(f)

オブジェクトプールインスタンス変数. clearValue(伝票項目名)
オブジェクトプールインスタンス変数. clearDBValue(DB項目名)
オブジェクトプールインスタンス変数. clearAphandleValue(クラス間インターフェース項目名)
オブジェクトプールインスタンス変数. clearApconstValue(アプリケーション定数項目名)
オブジェクトプールインスタンス変数. clearProcessorValue(プロセッサ予約項目名)

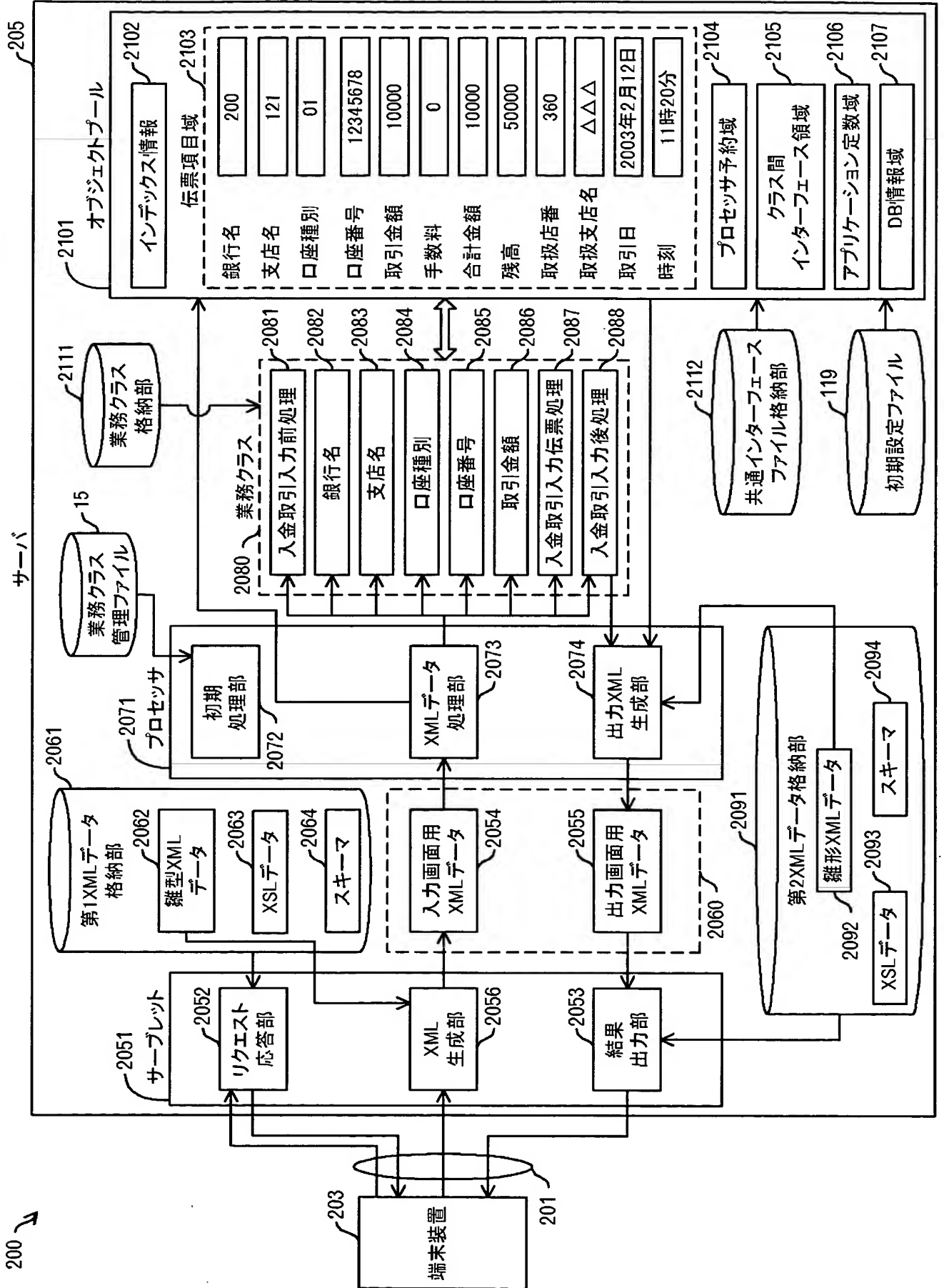
(g)

オブジェクトプールインスタンス変数. getAttributeVal(伝票項目名, 配列番号, 属性名)
--

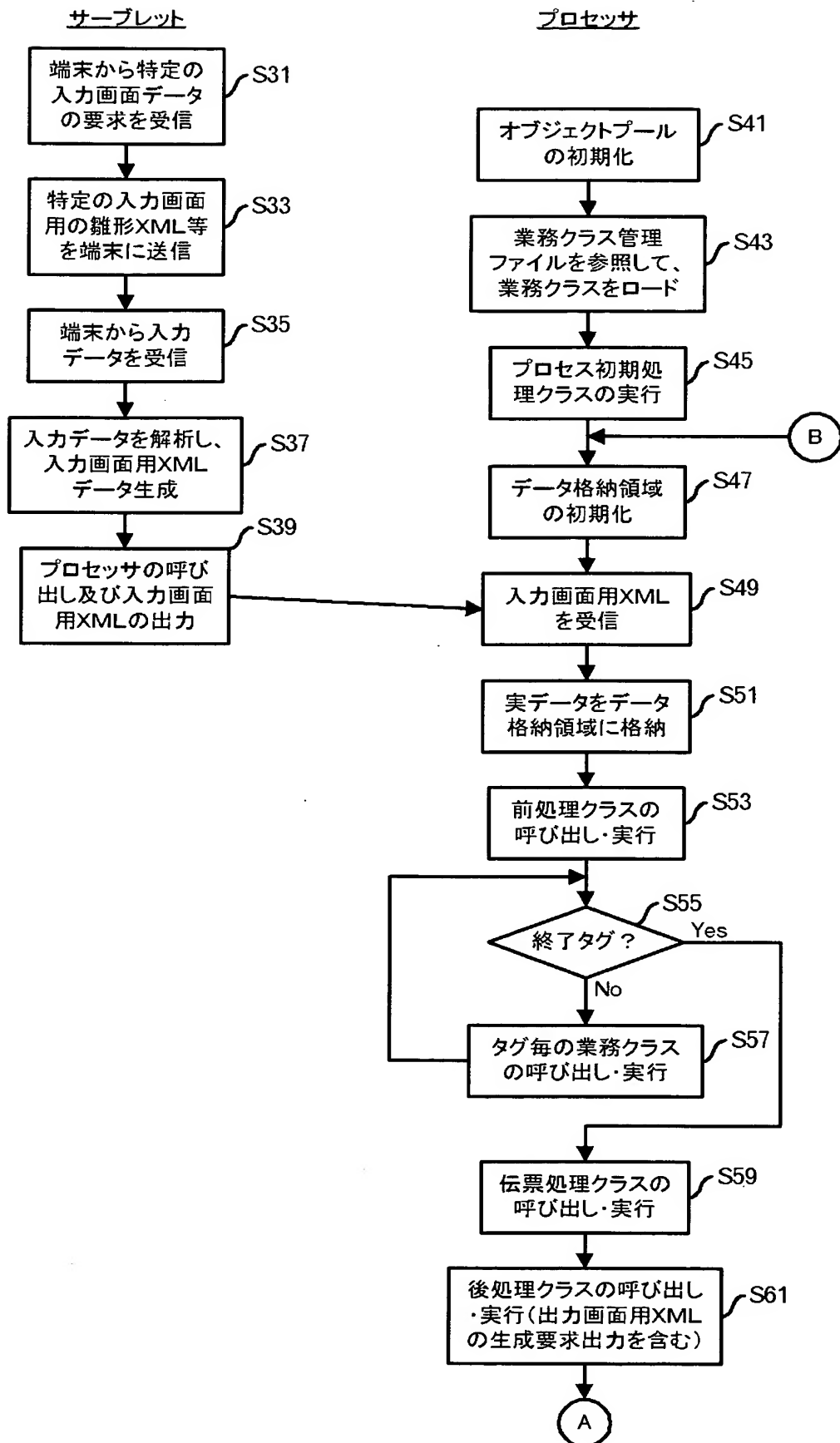
(h)

オブジェクトプールインスタンス変数. setAttributeVal(伝票項目名, 配列番号, 属性名, 属性値)
---

【図6】



【図 7】





【図 8】

入金取引(入力)

銀行名	〇〇〇	▼	601
支店名	× × ×	▼	602
口座種別	● 普通   ○ 定期   ○ 当座   603		
口座番号	12345678		604
取引金額	10000		605
確認		戻る	606   607

【図 9】

&lt;入金取引入力&gt;

&lt;銀行名&gt;200&lt;/銀行名&gt;

&lt;支店名&gt;121&lt;/支店名&gt;

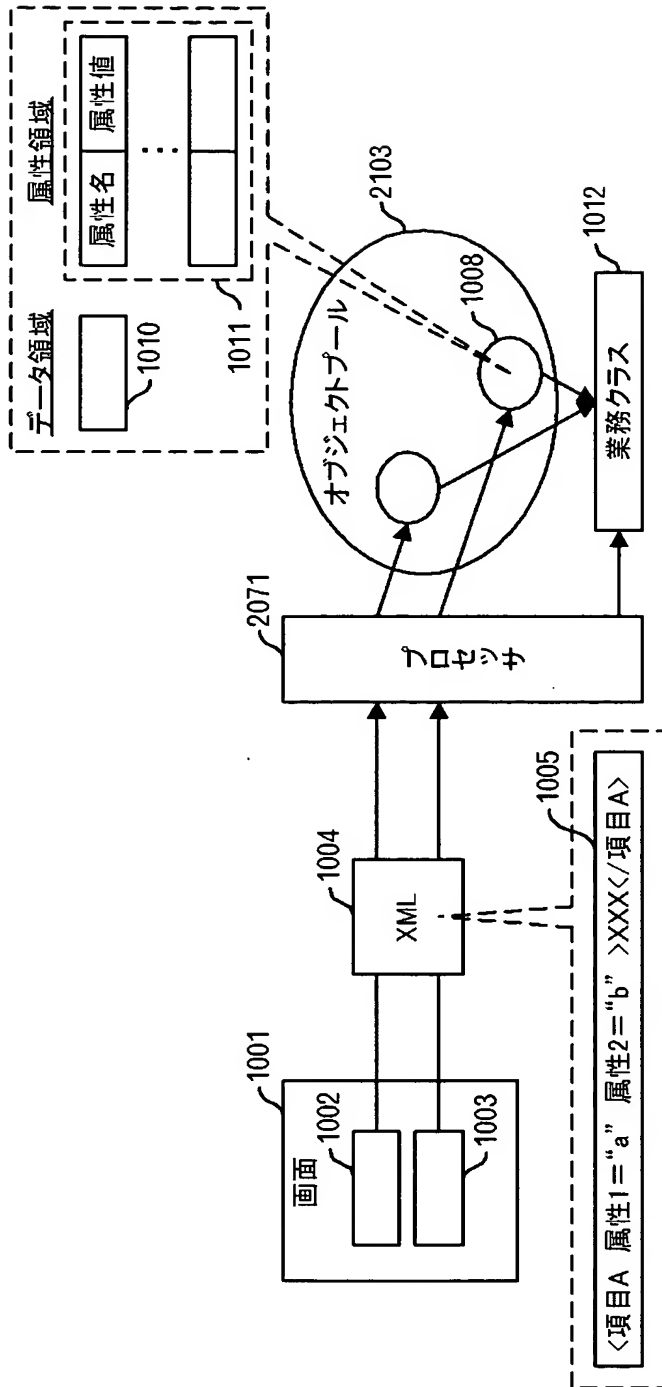
&lt;口座種別&gt;01&lt;/口座種別&gt;

&lt;口座番号&gt;12345678&lt;/口座番号&gt;

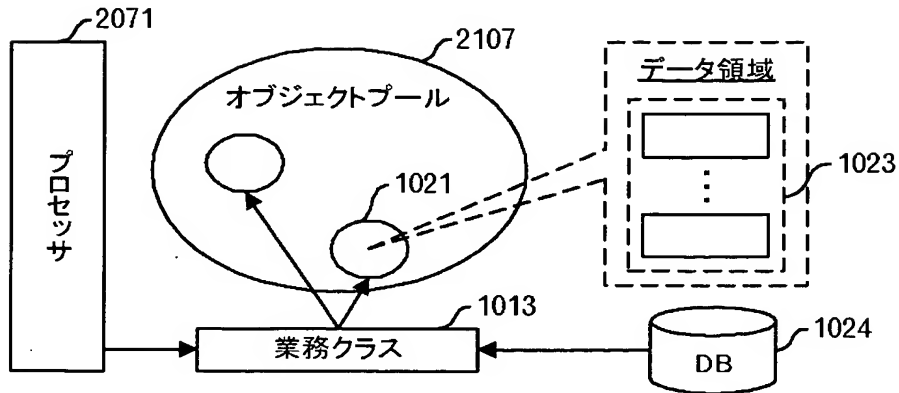
&lt;取引金額&gt;10000&lt;/取引金額&gt;

&lt;/入金取引入力&gt;

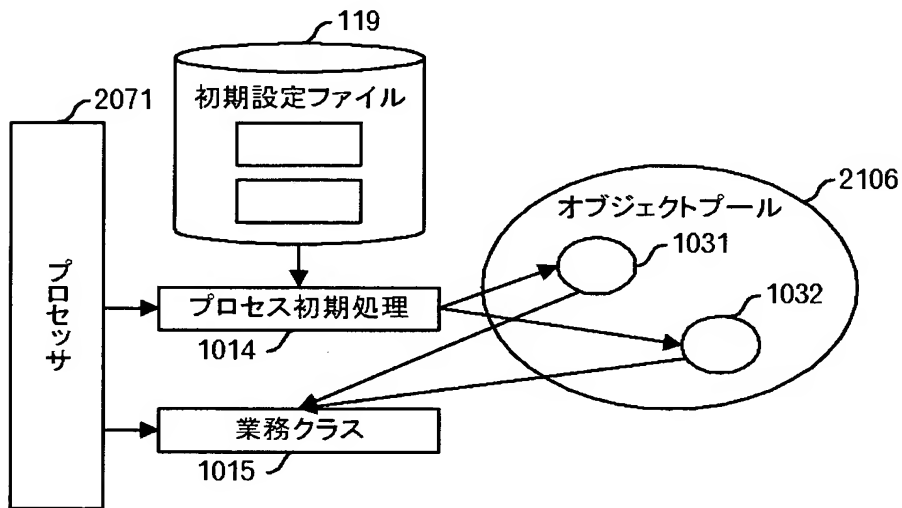
【図 10 A】



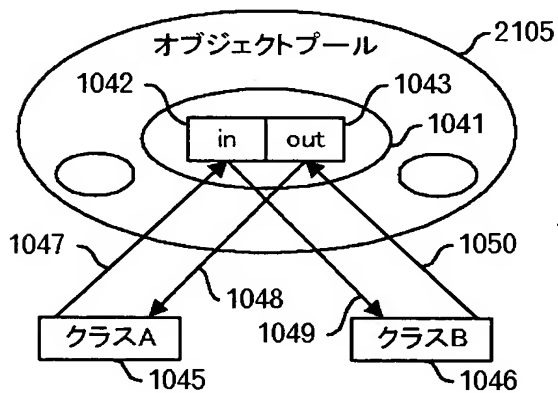
【図 10 B】



【図 10 C】



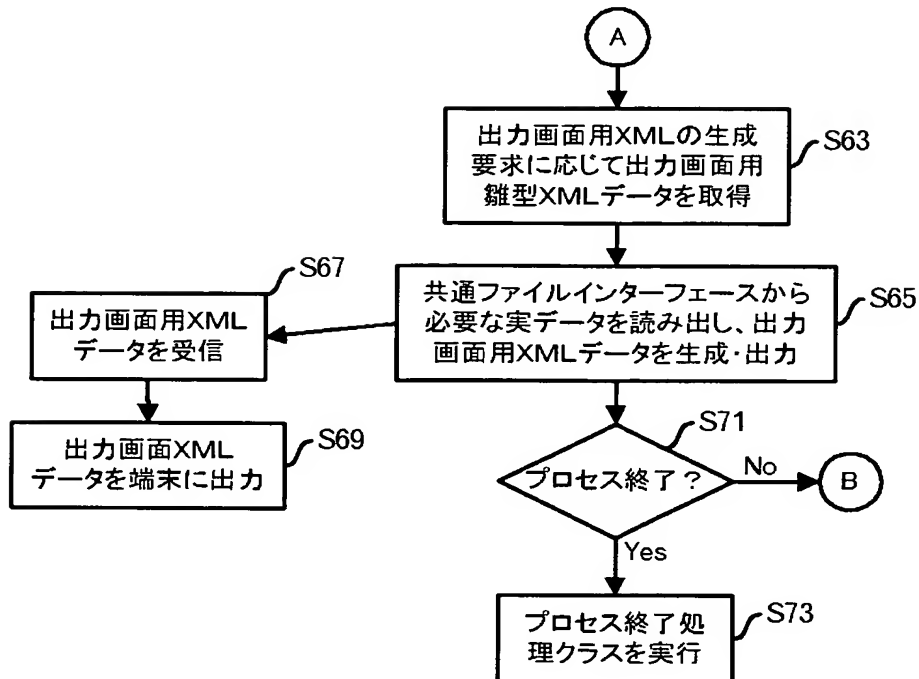
【図 10 D】



【図 1 1】

サーバ

プロセッサ



【図 1 2】

```

<?xml:stylesheet...出力画面. xsl?>
<入金取引結果>
  <取引日>2003年2月12日</取引日>
  <時刻>11時20分</時刻>
  <銀行名>〇〇〇</銀行名>
  <支店名>×××</支店名>
  <口座種別>普通</口座種別>
  <口座番号>12345678</口座番号>
  <取引金額>10000</取引金額>
  <手数料>0</手数料>
  <合計金額>10000</合計金額>
  <残高>50000</残高>
  <取扱店番>360</取扱店番>
  <取扱支店名>△△△</取扱支店名>
</入金取引結果>
    
```

【図 1 3】

入金取引(結果)	
取引日: 2003/2/12	時刻: 11時20分
銀行名: ○○○	支店名: × × ×
口座種別: 普通	口座番号: 12345678
取引金額: 10000	手数料: 0
合計金額: 10000	残高: 50000
取扱店番: 360	取扱支店名: △△△
1301 確認	1302 初画面

【書類名】 要約書

【要約】

【課題】

XMLドリブンアーキテクチャの業務プログラムの開発を行う。

【解決手段】

まず、XMLデータ格納部に格納され且つ帳票画面に対応するXMLデータを解析し、XMLデータに含まれるタグに対応する業務クラスを特定し、生成すべき業務クラスが登録された業務クラス管理部を参照して、特定された業務クラスが未登録であるか判断し、特定された業務クラスが未登録である場合には、業務クラス管理部に当該特定された業務クラスを登録する。この後各業務クラスの具体的なコーディングを行う。一方、作成された業務クラスについては、帳票画面への入力又は選択データに対応するタグを含むXMLデータをメモリに格納し、当該XMLデータに含まれるタグに対応し且つ当該タグに関連する処理を実施するためのプログラムである業務クラスを特定し、予め定義され且つメモリにロードされた業務クラスの中で、特定された業務クラスを呼び出すことにより使用する。

【選択図】 図 1

特願 2 0 0 4 - 0 2 5 6 7 7

出 願 人 履 歴 情 報

識別番号

[ 0 0 0 0 0 5 2 2 3 ]

1. 変更年月日

1 9 9 6 年 3 月 2 6 日

[変更理由]

住所変更

住 所

神奈川県川崎市中原区上小田中 4 丁目 1 番 1 号

氏 名

富士通株式会社